

Arkusz Kalkulacyjny

Kamil Czop
Projekt Programowanie Obiektowe

1 Indeks hierarchiczny	1
1.1 Hierarchia klas	1
2 Indeks klas	3
2.1 Lista klas	3
3 Indeks plików	5
3.1 Lista plików	5
4 Dokumentacja klas	7
4.1 Dokumentacja struktury BadFileException	7
4.1.1 Opis szczegółowy	8
4.2 Dokumentacja klasy Cell	8
4.2.1 Opis szczegółowy	10
4.2.2 Dokumentacja funkcji składowych	10
4.2.2.1 operator+()	10
4.2.2.2 operator<()	11
4.2.2.3 operator>()	11
4.3 Dokumentacja klasy Column	12
4.3.1 Opis szczegółowy	13
4.3.2 Dokumentacja konstruktora i destruktor	13
4.3.2.1 Column()	13
4.3.3 Dokumentacja funkcji składowych	14
4.3.3.1 begin()	14
4.3.3.2 end()	14
4.3.3.3 generateCellArray()	14
4.3.3.4 getCell()	15
4.3.3.5 getHeight()	16
4.3.3.6 getType()	16
4.3.3.7 operator[]()	16
4.3.3.8 resize()	17
4.4 Dokumentacja klasy IntCell	18
4.4.1 Opis szczegółowy	21
4.4.2 Dokumentacja konstruktora i destruktor	21
4.4.2.1 IntCell() [1/2]	21
4.4.2.2 IntCell() [2/2]	21
4.4.3 Dokumentacja funkcji składowych	22
4.4.3.1 getCalcValue()	22
4.4.3.2 getValue()	22
4.4.3.3 setValue()	22
4.5 Dokumentacja struktury NotNumericValue	23
4.5.1 Opis szczegółowy	24
4.6 Dokumentacja klasy NumericCell	24

4.6.1 Opis szczegółowy	26
4.6.2 Dokumentacja funkcji składowych	27
4.6.2.1 getCalcValue()	27
4.6.2.2 operator+()	27
4.6.2.3 operator<()	28
4.6.2.4 operator>()	29
4.7 Dokumentacja klasy Sheet	29
4.7.1 Opis szczegółowy	30
4.7.2 Dokumentacja konstruktora i destruktora	30
4.7.2.1 Sheet()	31
4.7.3 Dokumentacja funkcji składowych	31
4.7.3.1 createColumnArray()	31
4.7.3.2 getColumn()	32
4.7.3.3 getHeight()	33
4.7.3.4 getWidth()	34
4.7.3.5 operator[]()	34
4.7.3.6 resize()	35
4.8 Dokumentacja klasy StringCell	36
4.8.1 Opis szczegółowy	39
4.8.2 Dokumentacja konstruktora i destruktora	39
4.8.2.1 StringCell()	39
4.8.3 Dokumentacja funkcji składowych	39
4.8.3.1 getValue()	39
4.8.3.2 operator+()	39
4.8.3.3 operator<()	40
4.8.3.4 operator>()	40
4.8.3.5 setValue()	41
4.9 Dokumentacja klasy StringInterface	41
4.9.1 Opis szczegółowy	43
4.9.2 Dokumentacja funkcji składowych	43
4.9.2.1 getValue()	43
4.9.2.2 setValue()	44
5 Dokumentacja plików	45
5.1 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/cell.hpp	45
5.2 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/cellType.hpp	46
5.2.1 Dokumentacja typów wyliczanych	46
5.2.1.1 CellType	46
5.3 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/intCell.cpp	47
5.4 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/intCell.hpp	47
5.5 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/numericCell.cpp	49
5.6 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/numericCell.hpp	49

5.7 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/stringCell.cpp	51
5.8 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/stringCell.hpp	51
5.9 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp	52
5.10 Dokumentacja pliku ProgramowanieObiektowe/arkusz/column/column.cpp	53
5.11 Dokumentacja pliku ProgramowanieObiektowe/arkusz/column/column.hpp	54
5.12 Dokumentacja pliku ProgramowanieObiektowe/arkusz/tablica/tablica.cpp	55
5.13 Dokumentacja pliku ProgramowanieObiektowe/arkusz/tablica/tablica.hpp	56
5.14 Dokumentacja pliku ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.cpp	56
5.14.1 Dokumentacja funkcji	57
5.14.1.1 DisplaySheet()	57
5.15 Dokumentacja pliku ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.hpp	58
5.15.1 Dokumentacja funkcji	59
5.15.1.1 DisplaySheet()	60
5.16 Dokumentacja pliku ProgramowanieObiektowe/error.hpp	60
5.16.1 Dokumentacja typów wyliczanych	61
5.16.1.1 Wyjatki	62
5.17 Dokumentacja pliku ProgramowanieObiektowe/io/fileOperations.cpp	62
5.17.1 Dokumentacja funkcji	63
5.17.1.1 loadFile()	63
5.17.1.2 saveFile()	63
5.18 Dokumentacja pliku ProgramowanieObiektowe/io/fileOperations.hpp	64
5.18.1 Dokumentacja funkcji	65
5.18.1.1 loadFile()	66
5.18.1.2 saveFile()	66
5.19 Dokumentacja pliku ProgramowanieObiektowe/main.cpp	67
5.20 Dokumentacja pliku ProgramowanieObiektowe/menu/menu.cpp	68
5.20.1 Dokumentacja funkcji	69
5.20.1.1 changeType()	70
5.20.1.2 columnParameters()	70
5.20.1.3 expandSheet()	71
5.20.1.4 generujMenu()	72
5.20.1.5 inputValue()	72
5.20.1.6 loadSheet()	73
5.20.1.7 obslugaMenu()	74
5.20.1.8 rowParameters()	75
5.20.1.9 saveSheet()	76
5.20.1.10 sheetCreator()	77
5.20.1.11 sheetParameters()	78
5.20.1.12 sort()	79
5.21 Dokumentacja pliku ProgramowanieObiektowe/menu/menu.hpp	80
5.21.1 Dokumentacja funkcji	81
5.21.1.1 changeType()	81

5.21.1.2 columnParameters()	82
5.21.1.3 expandSheet()	83
5.21.1.4 generujMenu()	84
5.21.1.5 inputValue()	84
5.21.1.6 loadSheet()	85
5.21.1.7 obslugaMenu()	86
5.21.1.8 rowParameters()	86
5.21.1.9 saveSheet()	87
5.21.1.10 sheetCreator()	88
5.21.1.11 sheetParameters()	89
5.21.1.12 sort()	90
5.22 Dokumentacja pliku ProgramowanieObiektowe/operacje/operacje.cpp	90
5.22.1 Dokumentacja funkcji	92
5.22.1.1 countCalculateableColumns()	92
5.22.1.2 maxColumn()	92
5.22.1.3 maxRow()	93
5.22.1.4 minColumn()	94
5.22.1.5 minRow()	94
5.22.1.6 sortColumn()	95
5.22.1.7 sumColumn()	96
5.22.1.8 sumRow()	97
5.23 Dokumentacja pliku ProgramowanieObiektowe/operacje/operacje.hpp	97
5.23.1 Dokumentacja funkcji	99
5.23.1.1 countCalculateableColumns()	99
5.23.1.2 maxColumn()	100
5.23.1.3 maxRow()	100
5.23.1.4 minColumn()	101
5.23.1.5 minRow()	102
5.23.1.6 sortColumn()	102
5.23.1.7 sumColumn()	103
5.23.1.8 sumRow()	104
5.24 Dokumentacja pliku ProgramowanieObiektowe/utility/utility.cpp	105
5.24.1 Dokumentacja funkcji	105
5.24.1.1 czyscBufor()	105
5.24.1.2 wprowadzZakres()	106
5.25 Dokumentacja pliku ProgramowanieObiektowe/utility/utility.hpp	107
5.25.1 Dokumentacja funkcji	108
5.25.1.1 czyscBufor()	108
5.25.1.2 wprowadzZakres()	109

Chapter 1

Indeks hierarchiczny

1.1 Hierarchia klas

Ta lista dziedziczenia posortowana jest z grubsza, choć nie całkowicie, alfabetycznie:

Column	12
std::exception	
BadFileException	7
NotNumericValue	23
Sheet	29
StringInterface	41
Cell	8
NumericCell	24
IntCell	18
StringCell	36

Chapter 2

Indeks klas

2.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

BadFileException	Wyjątek wyrzucany w przypadku braku dostępu. Wyjątek jest wyrzucany gdy użytkownik nie ma dostępu do pliku lub nieistnieje (odczyt)	7
Cell	Interfejs/ klasa czysto wirtualna komórka Interfejs komórka posiadający metody ogólne klas dziedziczących	8
Column	Klasa określająca kolumnę Klasa kolumna obsługująca podstawowe funkcjonalności kolekcji posiada wskaźnik początku i końca zakresu celem wykorzystania w funkcjach z biblioteki algorithm czy pętlach zakresowych	12
IntCell	IntCell komórka z wartością całkowitą Komórka przyjmująca wartości całkowite	18
NotNumericValue	Wyjątek wyrzucany w przypadku wprowadzenia wartości nie numerycznej do komórki. Wyjątek powinien być wyrzucany gdy wprowadzony element nie może być przetworzony na wartość liczbowa	23
NumericCell	NumericCell komórki liczbowe Ogólna klasa abstrakcyjna komórek liczbowych które mogą posługiwać się wartościami rzeczywistymi	24
Sheet	Klasa opisująca Arkusz Klasa Arkusz przechowywująca tablicę kolumn i jej rozmiar	29
StringCell	StringCell Komórka tekstowa Klasa komórki przyjmującej wartości tekstowe	36
StringInterface	Intefejs elementów przyjmujących/zwracających elementy string	41

Chapter 3

Indeks plików

3.1 Lista plików

Tutaj znajduje się lista wszystkich udokumentowanych plików z ich krótkimi opisami:

ProgramowanieObiektowe/error.hpp	60
ProgramowanieObiektowe/main.cpp	67
ProgramowanieObiektowe/arkusz/cell/cell.hpp	45
ProgramowanieObiektowe/arkusz/cell/cellType.hpp	46
ProgramowanieObiektowe/arkusz/cell/intCell.cpp	47
ProgramowanieObiektowe/arkusz/cell/intCell.hpp	47
ProgramowanieObiektowe/arkusz/cell/numericCell.cpp	49
ProgramowanieObiektowe/arkusz/cell/numericCell.hpp	49
ProgramowanieObiektowe/arkusz/cell/stringCell.cpp	51
ProgramowanieObiektowe/arkusz/cell/stringCell.hpp	51
ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp	52
ProgramowanieObiektowe/arkusz/column/column.cpp	53
ProgramowanieObiektowe/arkusz/column/column.hpp	54
ProgramowanieObiektowe/arkusz/tablica/tablica.cpp	55
ProgramowanieObiektowe/arkusz/tablica/tablica.hpp	56
ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.cpp	56
ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.hpp	58
ProgramowanieObiektowe/io/fileOperations.cpp	62
ProgramowanieObiektowe/io/fileOperations.hpp	64
ProgramowanieObiektowe/menu/menu.cpp	68
ProgramowanieObiektowe/menu/menu.hpp	80
ProgramowanieObiektowe/operacje/operacje.cpp	90
ProgramowanieObiektowe/operacje/operacje.hpp	97
ProgramowanieObiektowe/utility/utility.cpp	105
ProgramowanieObiektowe/utility/utility.hpp	107

Chapter 4

Dokumentacja klas

4.1 Dokumentacja struktury BadFileException

Wyjątek wyrzucany w przypadku braku dostępu. Wyjątek jest wyrzucany gdy użytkownik nie ma dostępu do pliku lub nie istnieje (odczyt).

```
#include <error.hpp>
```

Diagram dziedziczenia dla BadFileException

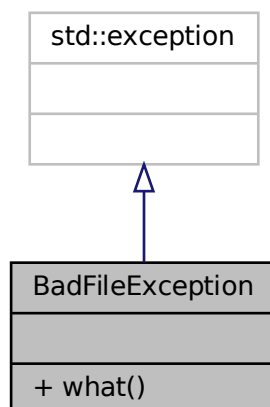
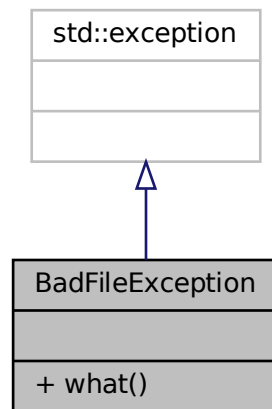


Diagram współpracy dla BadFileException:



Metody publiczne

- `const char * what () const throw ()`

4.1.1 Opis szczegółowy

Wyjątek wyrzucany w przypadku braku dostępu. Wyjątek jest wyrzucany gdy użytkownik nie ma dostępu do pliku lub nie istnieje (odczyt).

Dokumentacja dla tej struktury została wygenerowana z pliku:

- ProgramowanieObiektowe/[error.hpp](#)

4.2 Dokumentacja klasy Cell

Interfejs/ klasa czysto wirtualna komórka Interfejs komórka posiadający metody ogólne klas dziedziczących.

```
#include <cell.hpp>
```

Diagram dziedziczenia dla Cell

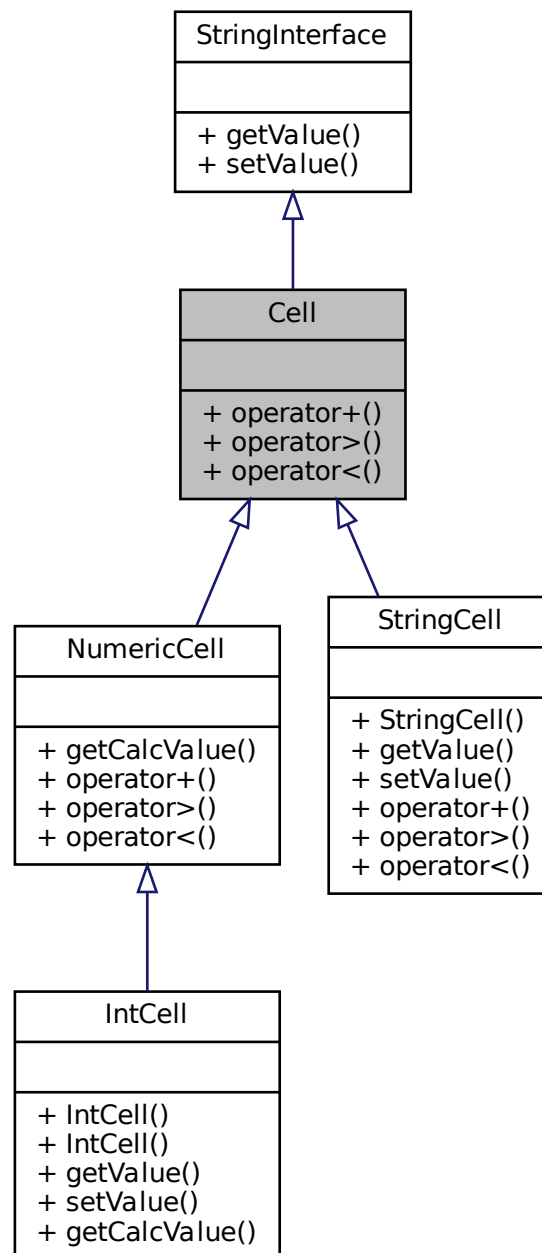
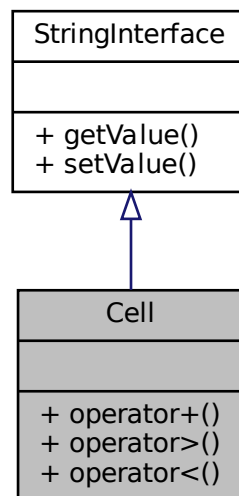


Diagram współpracy dla Cell:



Metody publiczne

- virtual double **operator+** (double rhs)=0
operator + Dodawanie wartości komórki z wartością double Przeciążenie operatora dodawania w przypadku wystąpienia wartości double przy sumie komórki
- virtual bool **operator>** (Cell &rhs)=0
operator > operator porównania Operator porównania czy obecny obiekt jest większy od drugiej komórki, Zwracana wartość w przypadku komórek które nie są ze sobą pokrewne w dziedziczeniu mogą się różnić.
- virtual bool **operator<** (Cell &rhs)=0
operator < operator porównania perator porównania Operator porównania czy obecny obiekt jest większy od drugiej komórki, Zwracana wartość w przypadku komórek które nie są ze sobą pokrewne w dziedziczeniu mogą się różnić.

4.2.1 Opis szczegółowy

Interfejs/ klasa czysto wirtualna komórka Interfejs komórka posiadający metody ogólne klas dziedziczących.

4.2.2 Dokumentacja funkcji składowych

4.2.2.1 operator+()

```
virtual double Cell::operator+ (
    double rhs ) [pure virtual]
```

operator + Dodawanie wartości komórki z wartością double Przeciążenie operatora dodawania w przypadku wystąpienia wartości double przy sumie komórki

Parametry

<i>in</i>	<i>rhs</i>	wartość którą sumujemy z komórką
-----------	------------	----------------------------------

Zwraca

sumę wartości

Implementowany w [StringCell](#) i [NumericCell](#).

4.2.2.2 operator<()

```
virtual bool Cell::operator< (  
    Cell & rhs ) [pure virtual]
```

operator < operator porównania perator porównania Operator porównania czy obecny obiekt jest większy od drugiej komórki, Zwracana wartość w przypadku komórek które nie są ze sobą pokrewne w dziedziczeniu mogą się różnić.

Parametry

<i>rhs</i>	komórka którą porównujemy
------------	---------------------------

Zwraca

Czy wartość obecnej komórki jest większa od komórki rhs

Implementowany w [StringCell](#) i [NumericCell](#).

4.2.2.3 operator>()

```
virtual bool Cell::operator> (  
    Cell & rhs ) [pure virtual]
```

operator > operator porównania Operator porównania czy obecny obiekt jest większy od drugiej komórki, Zwracana wartość w przypadku komórek które nie są ze sobą pokrewne w dziedziczeniu mogą się różnić.

Parametry

<i>rhs</i>	komórka którą porównujemy
------------	---------------------------

Zwraca

Czy wartość obecnej komórki jest większa od komórki rhs

Implementowany w [StringCell](#) i [NumericCell](#).

Dokumentacja dla tej klasy została wygenerowana z pliku:

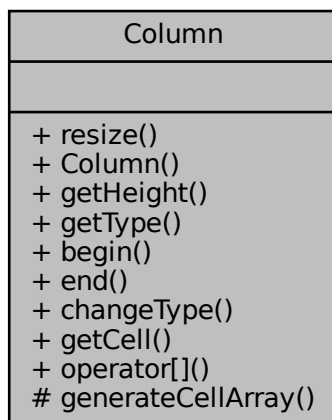
- [ProgramowanieObiektowe/arkusz/cell/cell.hpp](#)

4.3 Dokumentacja klasy Column

Klasa określająca kolumnę Klasa kolumna obsługująca podstawowe funkcjonalności kolekcji posiada wskaźnik początku i końca zakresu celem wykorzystania w funkcjach z biblioteki `algorithm` czy pętlach zakresowych.

```
#include <column.hpp>
```

Diagram współpracy dla Column:



Metody publiczne

- void [resize](#) (size_t newHeight)
resize metoda rozszerzająca kolumnę Metoda zajmuje się rozszerzaniem i zmniejszaniem obecnie przechowywanej tablicy Możliwa utrata danych przy zmienianiu rozmiaru na mniejszy
- [Column](#) (size_t height, [CellType](#) type)
[Column](#) Konstruktor kolumny o określonym rozmiarze i typie Tworzy nową kolumnę z tablicą o określonym rozmiarze na wskaźniki komórek określonego typu.
- std::size_t [getHeight](#) ()
Getter wysokości kolumny Zwraca rozmiar tablicy w kolumnie.
- [CellType](#) [getType](#) ()
getType Getter typu kolumny Zwraca typ komórek jaką kolumna przechowywuje
- [Cell](#) ** [begin](#) ()
begin Zwraca początek tablicy Metoda zwracają wskaźnik na początek tablicy komórek
- [Cell](#) ** [end](#) ()

end Zwraca koniec tablicy Metoda zwracająca wskaźnik na koniec tablicy komórek

- void **changeType** ([CellType](#) newType)
- [Cell](#) & **getCell** (size_t y)

getCell metoda zwraca referencje do komórki Metoda zwraca referencje do wybranej komórki jeśli jest ona w zakresie kolumny

- [Cell](#) & **operator[]** (size_t y)

getCell operator zwracający referencje do komórki Operator zwraca referencje do wybranej komórki jeśli jest ona w zakresie kolumny

Statyczne metody chronione

- static [Cell](#) ** **generateCellArray** (size_t height, [CellType](#) type)

Typ komórek w kolumnie.

4.3.1 Opis szczegółowy

Klasa określająca kolumnę Klasa kolumna obsługująca podstawowe funkcjonalności kolekcji posiada wskaźnik początku i końca zakresu celem wykorzystania w funkcjach z biblioteki `algorithm` czy pętlach zakresowych.

4.3.2 Dokumentacja konstruktora i destruktor

4.3.2.1 Column()

```
Column::Column (
    size_t height,
    CellType type )
```

[Column](#) Konstruktor kolumny o określonym rozmiarze i typie Tworzy nową kolumnę z tablicą o określonym rozmiarze na wskaźniki komórek określonego typu.

Parametry

in	<i>height</i>	Rozmiar tablicy komórek w kolumnie
in	<i>type</i>	Typ tworzonych komórek w kolumnie

Wyjątki

<code>std::bad_array_new_length</code>	w przypadku zerowego rozmiaru
--	-------------------------------

Oto graf wywołań dla tej funkcji:



4.3.3 Dokumentacja funkcji składowych

4.3.3.1 begin()

```
Cell ** Column::begin ( )
```

begin Zwraca początek tablicy Metoda zwracająca wskaźnik na początek tablicy komórek

Zwraca

początek wewnętrznej tablicy

4.3.3.2 end()

```
Cell ** Column::end ( )
```

end Zwraca koniec tablicy Metoda zwracająca wskaźnik na koniec tablicy komórek

Zwraca

koniec tablicy komórek

4.3.3.3 generateCellArray()

```
Cell ** Column::generateCellArray (
    size_t height,
    CellType type ) [static], [protected]
```

Typ komórek w kolumnie.

generateCellArray metoda tworząca nową tablicę komórek Statyczna metoda zajmująca się tworzeniem jednowymiarowej tablicy komórek określonego typu

Parametry

in	<i>height</i>	Wysokość nowej tablicy
in	<i>type</i>	Typ tworzonych komórek

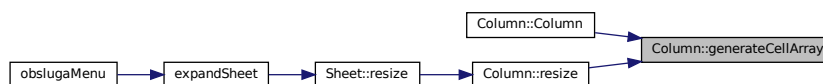
Wyjątki

<code>std::bad_array_new_length</code>	w przypadku zerowego rozmiaru
--	-------------------------------

Zwraca

Tablica jednowymiarowa wskaźników na komórki określonego typu

Oto graf wywoływań tej funkcji:



4.3.3.4 getCell()

```
Cell & Column::getCell (
    size_t y )
```

getCell metoda zwraca referencje do komórki Metoda zwraca referencje do wybranej komórki jeśli jest ona w zakresie kolumny

Parametry

in	<i>y</i>	współrzędna do komórki w tablicy
----	----------	----------------------------------

Wyjątki

<code>std::out_of_range</code>	Gdy y jest poza zakresem kolumny ($y > \text{height}$)
--------------------------------	--

Zwraca

referencja komórki z tablicy

Oto graf wywoływań tej funkcji:

**4.3.3.5 getHeight()**

```
size_t Column::getHeight ( )
```

Getter wysokości kolumny Zwraca rozmiar tablicy w kolumnie.

Zwraca

Rozmiar kolumny

4.3.3.6 getType()

```
CellType Column::getType ( )
```

getType Getter typu kolumny Zwraca typ komórek jaką kolumna przechowuje

Zwraca

Typ komórek w kolumnie

4.3.3.7 operator[]()

```
Cell & Column::operator[] (
    size_t y )
```

getCell operator zwracający referencje do komórki Operator zwraca referencje do wybranej komórki jeśli jest ona w zakresie kolumny

Parametry

in	y	współzędna do komórki w tablicy
----	---	---------------------------------

Wyjątki

<code>std::out_of_range</code>	Gdy y jest poza zakresem kolumny ($y > \text{height}$)
--------------------------------	--

Zwraca

referencja komórki z tablicy

Oto graf wywołań dla tej funkcji:



4.3.3.8 resize()

```
void Column::resize (
    size_t newHeight )
```

resize metoda rozszerzająca kolumnę Metoda zajmuje się rozszerzaniem i zmniejszaniem obecnie przechowywanej tablicy Możliwa utrata danych przy zmienianiu rozmiaru na mniejszy

Wyjątki

<code>std::bad_array_new_length</code>	w przypadku zerowego rozmiaru
--	-------------------------------

Parametry

in	<i>newHeight</i>	Nowy rozmiar kolumny
----	------------------	----------------------

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- ProgramowanieObiektowe/arkusz/column/[column.hpp](#)
- ProgramowanieObiektowe/arkusz/column/[column.cpp](#)

4.4 Dokumentacja klasy IntCell

IntCell komórka z wartością całkowitą Komórka przyjmująca wartości całkowite.

```
#include <intCell.hpp>
```


Diagram dziedziczenia dla IntCell

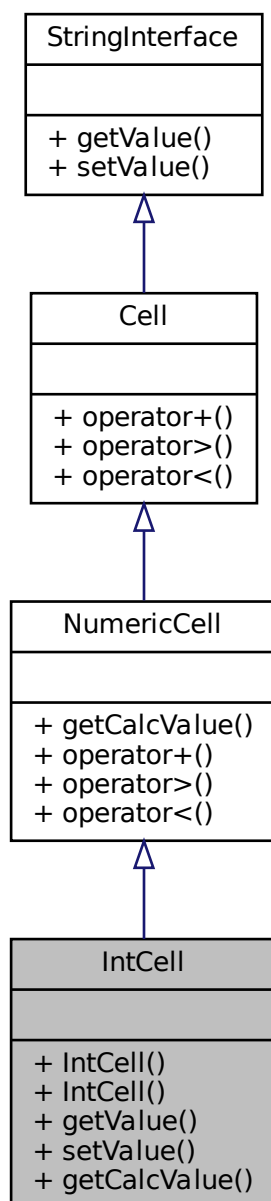
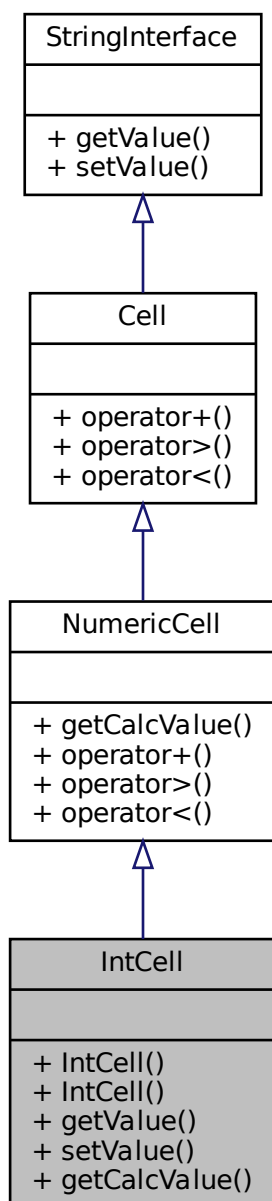


Diagram współpracy dla IntCell:



Metody publiczne

- **IntCell** (int value=0)

IntCell Konstruktor domyślny z opcjonalnym parametrem Konstruktor umożliwiający określenie początkowej wartości komórki.

- **IntCell** (std::string value)

IntCell Konstruktor z parametrem string Konstruktor umożliwiający określenie początkowej wartości komórki Wywołuje funkcję `setValue(std::string value)`

- `std::string getValue ()`
getValue Metoda ogólna od uzyskiwania wartości komórki Metoda zwraca wartość w postaci wartości tekstowej komórki
- `void setValue (std::string value)`
setValue Metoda ogólna od ustawiania wartości Ustawia wartość komórki za pomocą wartości string Wartość ta jest później parsowana W przypadku braku możliwości jej ustawienia wyrzucany jest wyjątek
- `double getCalcValue ()`
getCalcValue Metoda klasy umożliwiająca bezpośrednie uzyskanie wartości double Metoda umożliwia uzyskanie wartości bezpośrednie bez potrzeby parsowania wartości

4.4.1 Opis szczegółowy

`IntCell` komórka z wartością całkowitą Komórka przyjmująca wartości całkowite.

4.4.2 Dokumentacja konstruktora i destruktora

4.4.2.1 IntCell() [1/2]

```
IntCell::IntCell (
    int value = 0 )
```

`IntCell` Konstruktor domyślny z opcjonalnym parametrem Konstruktor umożliwiający określenie początkowej wartości komórki.

Parametry

in	value	Wartość początkowa komórki
----	-------	----------------------------

4.4.2.2 IntCell() [2/2]

```
IntCell::IntCell (
    std::string value )
```

`IntCell` Konstruktor z parametrem string Konstruktor umożliwiający określenie początkowej wartości komórki Wywołuje funkcję `setValue(std::string value)`

Parametry

in	value	Wartość początkowa komórki
----	-------	----------------------------

4.4.3 Dokumentacja funkcji składowych

4.4.3.1 getCalcValue()

```
double IntCell::getCalcValue ( ) [virtual]
```

getCalcValue Metoda klasy umożliwiająca bezpośrednie uzyskanie wartości double Metoda umożliwia uzyskanie wartości bezpośrednie bez potrzeby parsowania wartości

Zwraca

wartość komórki

Implementuje [NumericCell](#).

4.4.3.2 getValue()

```
std::string IntCell::getValue ( ) [virtual]
```

getValue Metoda ogólna od uzyskiwania wartości komórki Metoda zwraca wartość w postaci wartości tekstowej komórki

Zwraca

zawartość tekstowa komórki

Implementuje [StringInterface](#).

4.4.3.3 setValue()

```
void IntCell::setValue (
    std::string value ) [virtual]
```

setValue Metoda ogólna od ustawiania wartości Ustawia wartość komórki za pomocą wartości string Wartość ta jest później parsowana W przypadku braku możliwości jej ustawienia wyrzucany jest wyjątek

Wyjątki

NotNumericValue	Brak możliwości przetworzenia wartości tekstowej na liczbową
---------------------------------	--

Parametry

in	value	ustawiana wartość
----	-------	-------------------

Implementuje [StringInterface](#).

Dokumentacja dla tej klasy została wygenerowana z plików:

- ProgramowanieObiektowe/arkusz/cell/[intCell.hpp](#)
- ProgramowanieObiektowe/arkusz/cell/[intCell.cpp](#)

4.5 Dokumentacja struktury NotNumericValue

Wyjątek wyrzucany w przypadku wprowadzenia wartości nie numerycznej do komórki. Wyjątek powinien być wyrzucany gdy wprowadzony element nie może być przetworzony na wartość liczbową.

```
#include <error.hpp>
```

Diagram dziedziczenia dla NotNumericValue

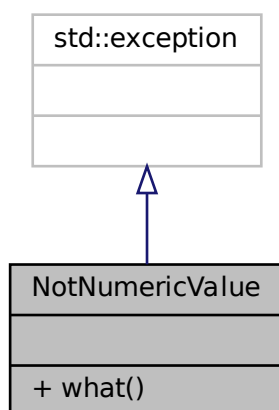
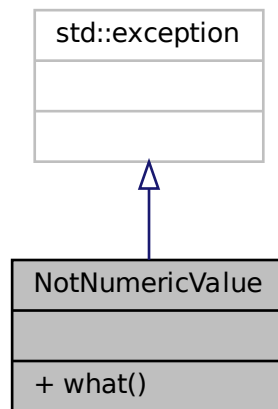


Diagram współpracy dla NotNumericValue:



Metody publiczne

- `const char * what () const throw ()`

4.5.1 Opis szczegółowy

Wyjątek wyrzucany w przypadku wprowadzenia wartości nie numerycznej do komórki. Wyjątek powinien być wyrzucany gdy wprowadzony element nie może być przetworzony na wartość liczbową.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- ProgramowanieObiektowe/[error.hpp](#)

4.6 Dokumentacja klasy NumericCell

[NumericCell](#) komórki liczbowe Ogólna klasa abstrakcyjna komórek liczbowych które mogą posługiwać się wartościami rzeczywistymi.

```
#include <numericCell.hpp>
```

Diagram dziedziczenia dla NumericCell

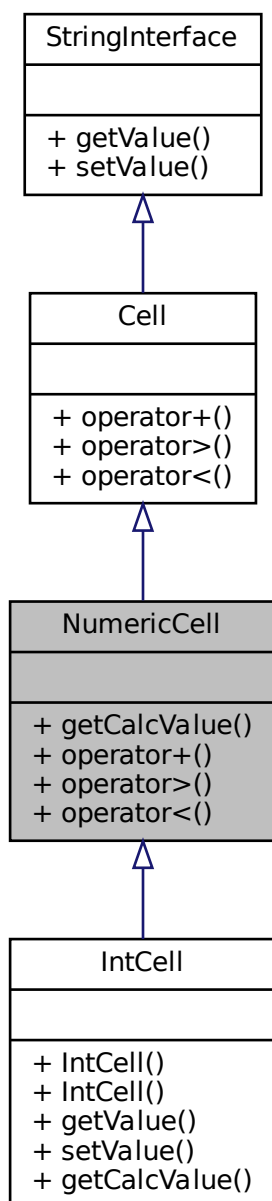
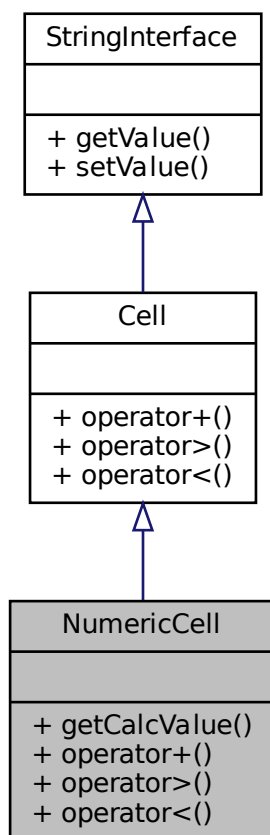


Diagram współpracy dla NumericCell:



Metody publiczne

- virtual double `getCalcValue ()`=0
getCalcValue Metoda klasy umożliwiająca bezpośrednie uzyskanie wartości double Metoda umożliwia uzyskanie wartości bezpośrednio bez potrzeby parsowania wartości
- double `operator+` (double rhs)
operator + Dodawanie wartości komórki z wartością double Przeciążenie operatora dodawania w przypadku wystąpienia wartości double przy sumie komórki
- bool `operator>` (Cell &)
operator > operator porównania Operator porównania czy obecny obiekt jest większy od drugiej komórki, Zwracana wartość w przypadku komórek które nie dziedziczą po RealCell porównanie zwraca wartość FALSE
- bool `operator<` (Cell &)
operator < operator porównania Operator porównania czy obecny obiekt jest mniejszy od drugiej komórki, Zwracana wartość w przypadku komórek które nie dziedziczą po RealCell porównanie zwraca wartość FALSE

4.6.1 Opis szczegółowy

NumericCell komórki liczbowe Ogólna klasa abstrakcyjna komórek liczbowych które mogą posługiwać się wartościami rzeczywistymi.

4.6.2 Dokumentacja funkcji składowych

4.6.2.1 getCalcValue()

```
virtual double NumericCell::getCalcValue ( ) [pure virtual]
```

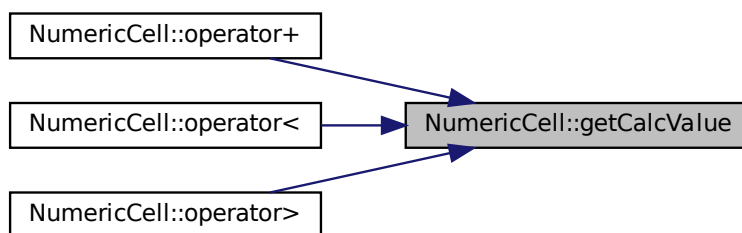
getCalcValue Metoda klasy umożliwiająca bezpośrednie uzyskanie wartości double Metoda umożliwia uzyskanie wartości bezpośrednio bez potrzeby parsowania wartości

Zwraca

wartość komórki

Implementowany w [IntCell](#).

Oto graf wywoływań tej funkcji:



4.6.2.2 operator+()

```
double NumericCell::operator+ (
    double rhs ) [virtual]
```

operator + Dodawanie wartości komórki z wartością double Przeciążenie operatora dodawania w przypadku wystąpienia wartości double przy sumie komórki

Parametry

in	rhs	wartość którą sumujemy z komórką
----	-----	----------------------------------

Zwraca

sumę wartości

Implementuje [Cell](#).

Oto graf wywołań dla tej funkcji:

**4.6.2.3 operator<()**

```
bool NumericCell::operator< (  
    Cell & rhs ) [virtual]
```

operator < operator porównania Operator porównania czy obecny obiekt jest mniejszy od drugiej komórki, Zwracana wartość w przypadku komórek które nie dziedziczą po RealCell porównanie zwraca wartość FALSE

Parametry

<i>rhs</i>	komórka którą porównujemy
------------	---------------------------

Zwraca

Czy wartość obecnej komórki jest większa od komórki rhs, jeśli komórka nie jest typu RealCell zwraca false

Implementuje [Cell](#).

Oto graf wywołań dla tej funkcji:



4.6.2.4 operator>()

```
bool NumericCell::operator> (
    Cell & rhs ) [virtual]
```

operator > operator porównania Operator porównania czy obecny obiekt jest większy od drugiej komórki, Zwracana wartość w przypadku komórek które nie dziedziczą po RealCell porównanie zwraca wartość FALSE

Parametry

<i>rhs</i>	komórka którą porównujemy
------------	---------------------------

Zwraca

Czy wartość obecnej komórki jest większa od komórki rhs, jeśli komórka nie jest typu RealCell zwraca false

Implementuje [Cell](#).

Oto graf wywołań dla tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

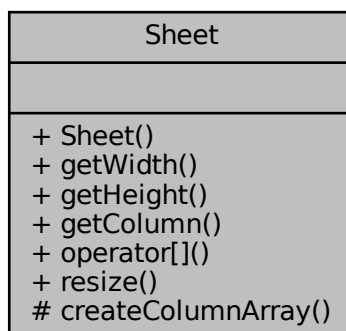
- [ProgramowanieObiektowe/arkusz/cell/numericCell.hpp](#)
- [ProgramowanieObiektowe/arkusz/cell/numericCell.cpp](#)

4.7 Dokumentacja klasy Sheet

Klasa opisująca Arkusz Klasa Arkusz przechowująca tablicę kolumn i jej rozmiar.

```
#include <tablica.hpp>
```

Diagram współpracy dla Sheet:



Metody publiczne

- [Sheet](#) (size_t width, size_t height, [CellType](#) *types)
[Sheet](#) Konstruktor tworzący arkusz z tablicą o wyznaczonym rozmiarze Konstruktor tworzący arkusz z tablicą o wyznaczonej ilości kolumn określonego typu i wierszy.
- size_t [getWidth](#) ()
getWidth getter szerokości Zwraca ilość kolumn w arkuszu
- size_t [getHeight](#) ()
getHeight getter wysokości Zwraca ilość komórek w kolumnie kiedy arkusz był tworzony/rozszerzany
- [Column](#) & [getColumn](#) (size_t x)
getColumn Metoda zwracająca referencję na kolumnę Zwraca referencję na wybraną kolumnę z arkusza
- [Column](#) & [operator\[\]](#) (size_t x)
operator [] przeciążenie operatora[] celem uzyskiwania odrębnej kolumny
- void [resize](#) (size_t x, size_t y)
resize Metoda rozszerzania arkusza

Statyczne metody chronione

- static [Column](#) ** [createColumnArray](#) (size_t width, size_t height, [CellType](#) *types)
Wysokość tablicy - ilość komórek w utworzonych kolumnach.

4.7.1 Opis szczegółowy

Klasa opisująca Arkusz Klasa Arkusz przechowująca tablicę kolumn i jej rozmiar.

4.7.2 Dokumentacja konstruktora i destruktor

4.7.2.1 Sheet()

```
Sheet::Sheet (
    size_t width,
    size_t height,
    CellType * types )
```

Sheet Konstruktor tworzący arkusz z tablicą o wyznaczonym rozmiarze Konstruktor tworzący arkusz z tablicą o wyznaczonej ilości kolumn określonego typu i wierszy.

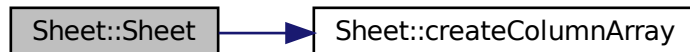
Parametry

in	<i>width</i>	Szerokość tablicy nowego arkusza
in	<i>height</i>	Wysokość nowej tablicy
in	<i>types</i>	Typy tworzonych kolumn

Wyjątki

<i>bad_array_new_length</i>	Zły rozmiar tworzonego arkusza
-----------------------------	--------------------------------

Oto graf wywołań dla tej funkcji:



4.7.3 Dokumentacja funkcji składowych

4.7.3.1 createColumnArray()

```
Column ** Sheet::createColumnArray (
    size_t width,
    size_t height,
    CellType * types ) [static], [protected]
```

Wysokość tablicy - ilość komórek w utworzonych kolumnach.

createColumnArray Tworzy nową dwuwymiarową tablicę. Funkcja generująca tablicę o określonym rozmiarze

Parametry

in	<i>width</i>	Szerokość nowej tablicy - ilość kolumn
in	<i>height</i>	Wysokość nowej tablicy - ilość komórek w kolumnach
in	<i>types</i>	Typy tworzonych kolumn

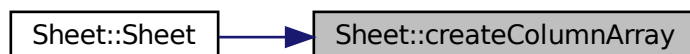
Wyjątki

<code>bad_array_new_length</code>	Zły rozmiar tworzonego arkusza
-----------------------------------	--------------------------------

Zwraca

Tworzy nową tablicę wskaźników kolumn o wyznaczonych rozmiarach

Oto graf wywołań tej funkcji:



4.7.3.2 getColumn()

```
Column & Sheet::getColumn (
    size_t x )
```

getColumn Metoda zwracająca referencję na kolumnę Zwraca referencję na wybraną kolumnę z arkusza

Parametry

in	x	Indeks kolumny
----	---	----------------

Wyjątki

<code>std::out_of_range</code>	Gdy x jest poza zakresem arkusza ($x > \text{width}$)
--------------------------------	---

Zwraca

Referencja na kolumnę

Oto graf wywoływań tej funkcji:



4.7.3.3 getHeight()

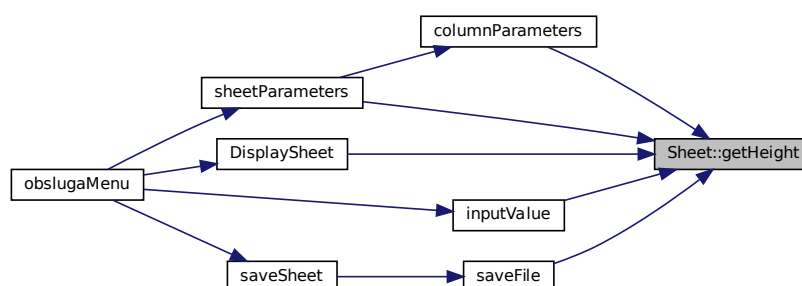
```
size_t Sheet::getHeight ( )
```

getHeight getter wysokości Zwraca ilość komórek w kolumnie kiedy arkusz był tworzony/rozszerzany

Zwraca

wysokość arkusza / ilość wierszy

Oto graf wywoływań tej funkcji:



4.7.3.4 getWidth()

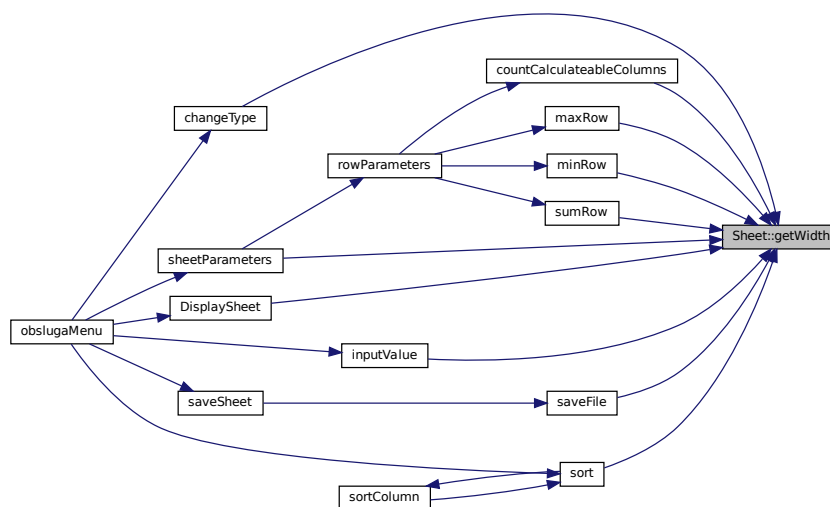
```
size_t Sheet::getWidth ( )
```

getWidth getter szerokości Zwraca ilość kolumn w arkuszu

Zwraca

ilość kolumn

Oto graf wywoływań tej funkcji:



4.7.3.5 operator[]()

```
Column & Sheet::operator[] (
    size_t x )
```

operator [] przeciążenie operatora[] celem uzyskiwania odrębnej kolumny

Zwraca referencję na wybraną kolumnę

Parametry

in	x	Indeks kolumny
----	---	----------------

Wyjątki

<code>std::out_of_range</code>	Gdy x jest poza zakresem arkusza ($x > \text{width}$)
--------------------------------	---

Zwraca

Referencja na wybraną kolumnę

Oto graf wywołań dla tej funkcji:

**4.7.3.6 resize()**

```
void Sheet::resize (
    size_t x,
    size_t y )
```

resize Metoda rozszerzania arkusza

Metoda zmienia rozmiar arkusza kopiując kolumny które także przechodzą zmianę rozmiaru. Nowe kolumny są automatycznie przeznaczone pod komórki typu [IntCell](#). Utrata danych w przypadku zmniejszania rozmiaru arkusza.

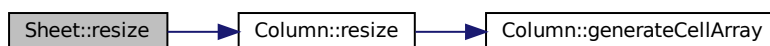
Parametry

in	x	Nowa szerokość arkusza
in	y	Nowa wysokość kolumn arkusza

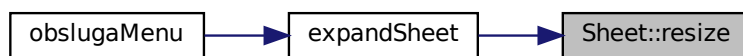
Wyjątki

<i>bad_array_new_length</i>	Zły rozmiar tworzonego arkusza
-----------------------------	--------------------------------

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- [ProgramowanieObiektowe/arkusz/tablica/tablica.hpp](#)
- [ProgramowanieObiektowe/arkusz/tablica/tablica.cpp](#)

4.8 Dokumentacja klasy StringCell

StringCell Komórka tekstowa Klasa komórki przyjmującej wartości tekstowe.

```
#include <stringCell.hpp>
```

Diagram dziedziczenia dla StringCell

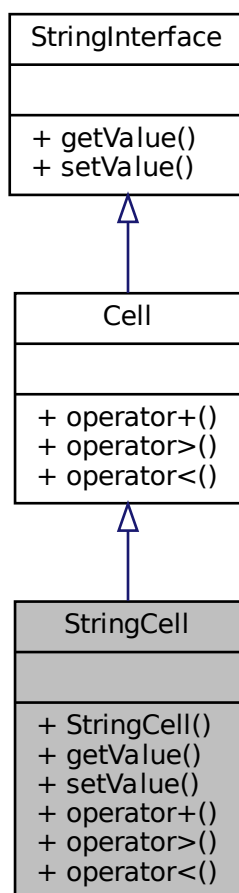
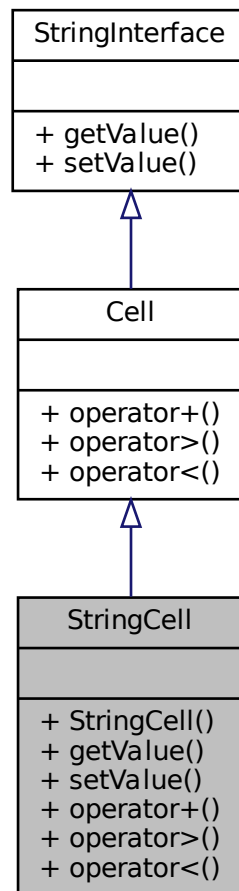


Diagram współpracy dla StringCell:



Metody publiczne

- **StringCell** (std::string value="?")
StringCell Konstruktor domyślny z opcjonalnym parametrem Konstruktor umożliwiający określenie początkowej wartości komórki.
- std::string **getValue** ()
getValue Metoda ogólna od uzyskiwania wartości komórki Metoda zwraca wartość w postaci wartości tekstowej komórki
- void **setValue** (std::string value)
setValue Metoda ogólna od ustawiania wartości Ustawia wartość komórki za pomocą wartości string
- double **operator+** (double rhs)
operator + Operator dodawania zwracający wartość rhs Operator dodawania w przypadku komórki która przyjmuje tylko wartości tekstowe zwraca domyślnie wartość wprowadzoną w parametrze operator+
- bool **operator>** (Cell &)
operator > Operator porównania perator porównania Operator porównania czy obecny obiekt jest większy od drugiej komórki, Zwracana wartość w przypadku komórek które nie są klasy *StringCell* uzyskiwana jest wartość true - komórka po prawej stronie będzie dominowała w porównaniu

- bool `operator<` (Cell &)

operator < operator porównania perator porównania Operator porównania czy obecny obiekt jest mniejszy od drugiej komórki, Zwracana wartość w przypadku komórek które nie są klasy [StringCell](#) uzyskiwana jest wartość true - komórka po prawej stronie będzie dominowała w porównaniu

4.8.1 Opis szczegółowy

[StringCell](#) Komórka tekstowa Klasa komórki przyjmującej wartości tekstowe.

4.8.2 Dokumentacja konstruktora i destruktora

4.8.2.1 StringCell()

```
StringCell::StringCell (
    std::string value = "?" )
```

[StringCell](#) Konstruktor domyślny z opcjonalnym parametrem Konstruktor umożliwiający określenie początkowej wartości komórki.

Parametry

in	value	Wartość początkowa komórki
----	-------	----------------------------

4.8.3 Dokumentacja funkcji składowych

4.8.3.1 getValue()

```
std::string StringCell::getValue ( ) [virtual]
```

getValue Metoda ogólna od uzyskiwania wartości komórki Metoda zwraca wartość w postaci wartości tekstowej komórki

Zwraca

zawartość tekstowa komórki

Implementuje [StringInterface](#).

4.8.3.2 operator+()

```
double StringCell::operator+ (
    double rhs ) [virtual]
```

operator + Operator dodawania zwracający wartość rhs Operator dodawania w przypadku komórki która przyjmuje tylko wartości tekstowe zwraca domyślnie wartość wprowadzoną w parametrze operator+

Parametry

<i>rhs</i>	Zwracana wartość
------------	------------------

Zwraca

Wartość wprowadzona w argumencie rhs

Implementuje [Cell](#).

4.8.3.3 operator<()

```
bool StringCell::operator< (
    Cell & rhs ) [virtual]
```

operator < operator porównania perator porównania Operator porównania czy obecny obiekt jest mniejszy od drugiej komórki, Zwracana wartość w przypadku komórek które nie są klasy [StringCell](#) uzyskiwana jest wartość true - komórka po prawej stronie będzie dominowała w porównaniu

Parametry

<i>rhs</i>	komórka którą porównujemy
------------	---------------------------

Zwraca

Czy wartość obecnej komórki jest większa od komórki rhs

Implementuje [Cell](#).

Oto graf wywołań dla tej funkcji:



4.8.3.4 operator>()

```
bool StringCell::operator> (
    Cell & rhs ) [virtual]
```

operator > operator porównania perator porównania Operator porównania czy obecny obiekt jest większy od drugiej komórki, Zwracana wartość w przypadku komórek które nie są klasy [StringCell](#) uzyskiwana jest wartość true - komórka po prawej stronie będzie dominowała w porównaniu

Parametry

<i>rhs</i>	komórka którą porównujemy
------------	---------------------------

Zwraca

Czy wartość obecnej komórki jest większa od komórki rhs

Implementuje [Cell](#).

Oto graf wywołań dla tej funkcji:



4.8.3.5 setValue()

```
void StringCell::setValue (
    std::string value ) [virtual]
```

setValue Metoda ogólna od ustawiania wartości Ustawia wartość komórki za pomocą wartości string

Parametry

<i>in</i>	<i>value</i>	ustawiana wartość
-----------	--------------	-------------------

Implementuje [StringInterface](#).

Dokumentacja dla tej klasy została wygenerowana z plików:

- ProgramowanieObiektowe/arkusz/cell/[stringCell.hpp](#)
- ProgramowanieObiektowe/arkusz/cell/[stringCell.cpp](#)

4.9 Dokumentacja klasy StringInterface

Intefejs elementów przyjmujących/zwracających elementy string.

```
#include <stringinterface.hpp>
```

Diagram dziedziczenia dla StringInterface

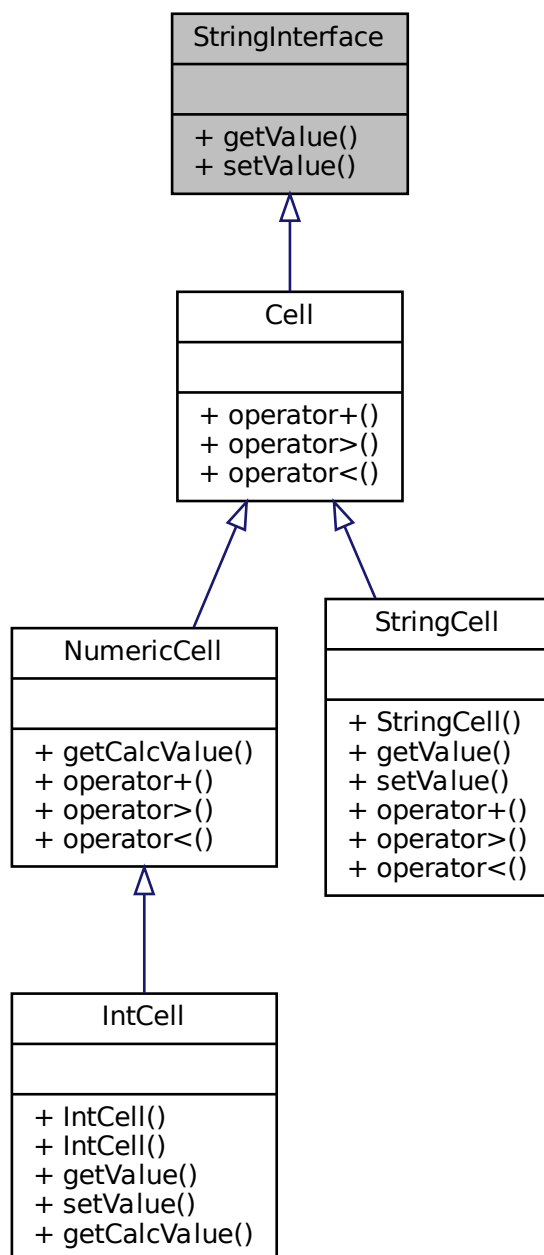
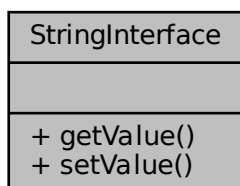


Diagram współpracy dla StringInterface:



Metody publiczne

- virtual std::string `getValue` ()=0
getValue Metoda ogólna od uzyskiwania wartości komórki Metoda zwraca wartość w postaci wartości tekstowej komórki
- virtual void `setValue` (std::string value)=0
setValue Metoda ogólna od ustawiania wartości Ustawia wartość komórki za pomocą wartości string

4.9.1 Opis szczegółowy

Intefejs elementów przyjmujących/zwracających elementy string.

4.9.2 Dokumentacja funkcji składowych

4.9.2.1 `getValue()`

```
virtual std::string StringInterface::getValue ( ) [pure virtual]
```

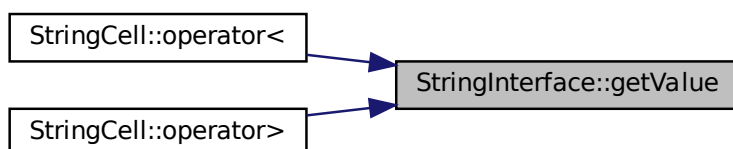
`getValue` Metoda ogólna od uzyskiwania wartości komórki Metoda zwraca wartość w postaci wartości tekstowej komórki

Zwraca

zawartość tekstowa komórki

Implementowany w [StringCell](#) i [IntCell](#).

Oto graf wywoływań tej funkcji:

**4.9.2.2 setValue()**

```
virtual void StringInterface::setValue (
    std::string value ) [pure virtual]
```

`setValue` Metoda ogólna od ustawiania wartości Ustawia wartość komórki za pomocą wartości string

Parametry

in	<i>value</i>	ustawiana wartość
----	--------------	-------------------

Implementowany w [StringCell](#) i [IntCell](#).

Dokumentacja dla tej klasy została wygenerowana z pliku:

- [ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp](#)

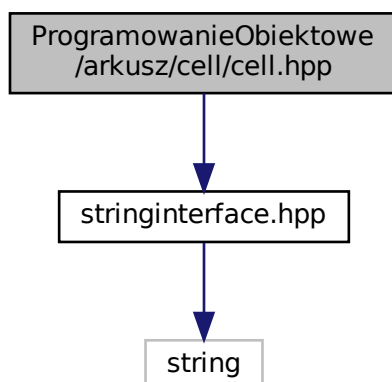
Chapter 5

Dokumentacja plików

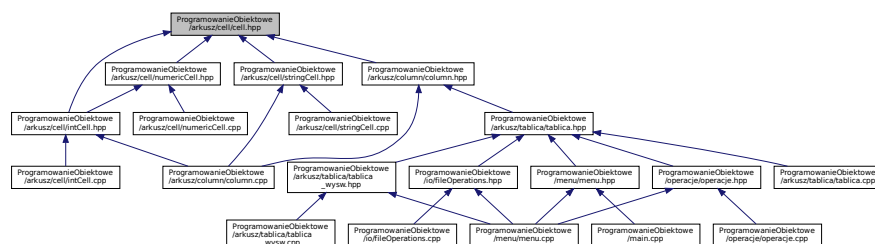
5.1 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/cell.hpp

```
#include "stringinterface.hpp"
```

Wykres zależności załączania dla cell.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

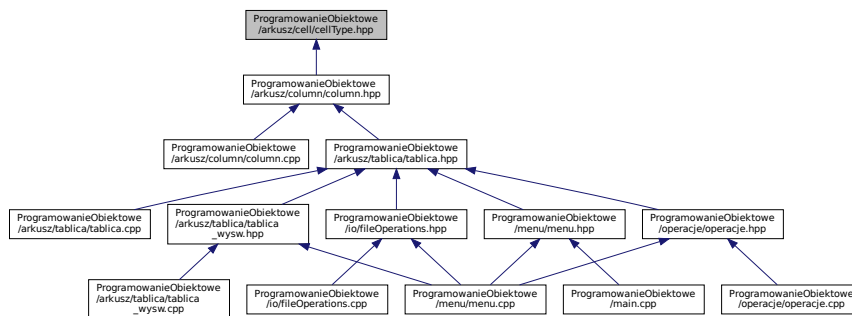
- class `Cell`

Interfejs/ klasa czysto wirtualna komórka Interfejs komórka posiadający metody ogólne klas dziedziczących.

5.2 Dokumentacja pliku

ProgramowanieObiektowe/arkusz/cell/cellType.hpp

Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Wyliczenia

- enum class `CellType` { `StringCell` = 0 , `IntCell` = 1 }

Typ wyliczeniowy typów komórek.

5.2.1 Dokumentacja typów wyliczanych

5.2.1.1 CellType

```
enum CellType [strong]
```

Typ wyliczeniowy typów komórek.

Wartości wyliczeń

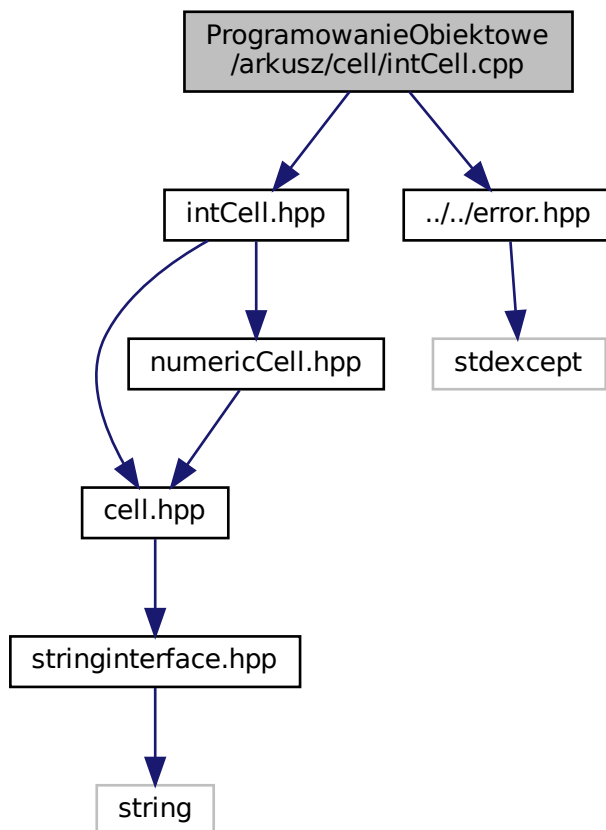
IntCell	Komórka tekstowa.
---------	-------------------

5.3 Dokumentacja pliku

ProgramowanieObiektowe/arkusz/cell/intCell.cpp

```
#include "intCell.hpp"  
#include "../error.hpp"
```

Wykres zależności załączania dla intCell.cpp:

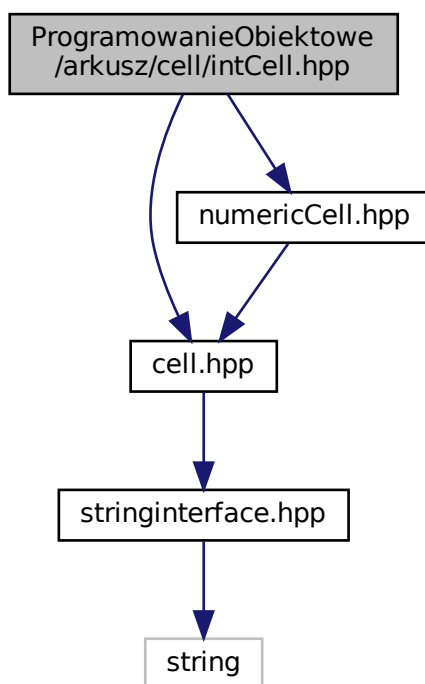


5.4 Dokumentacja pliku

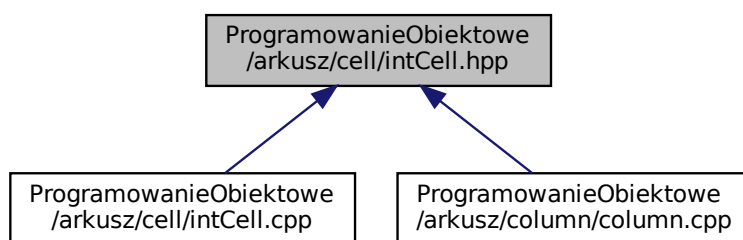
ProgramowanieObiektowe/arkusz/cell/intCell.hpp

```
#include "cell.hpp"  
#include "numericCell.hpp"
```

Wykres zależności załączania dla intCell.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

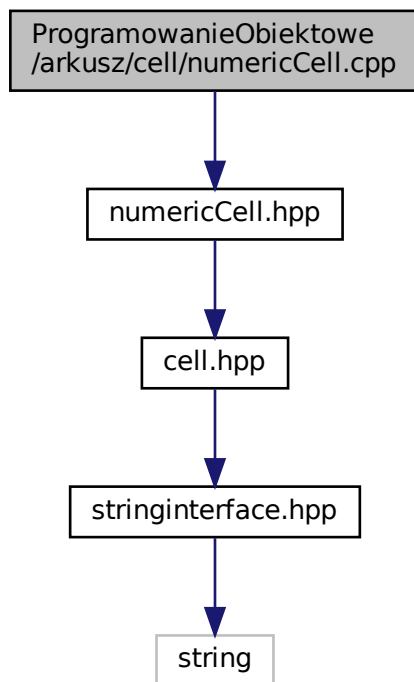
- class [IntCell](#)

[IntCell](#) komórka z wartością całkowitą Komórka przyjmująca wartości całkowite.

5.5 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/numericCell.cpp

```
#include "numericCell.hpp"
```

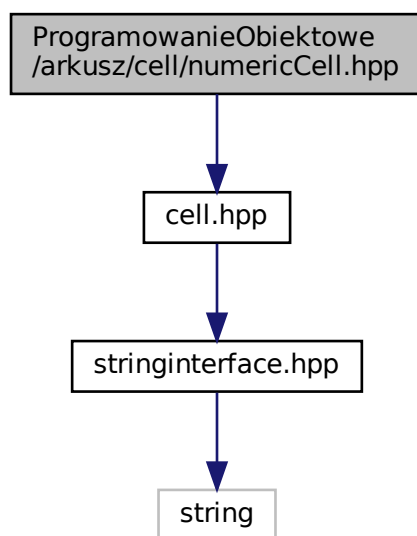
Wykres zależności załączania dla numericCell.cpp:



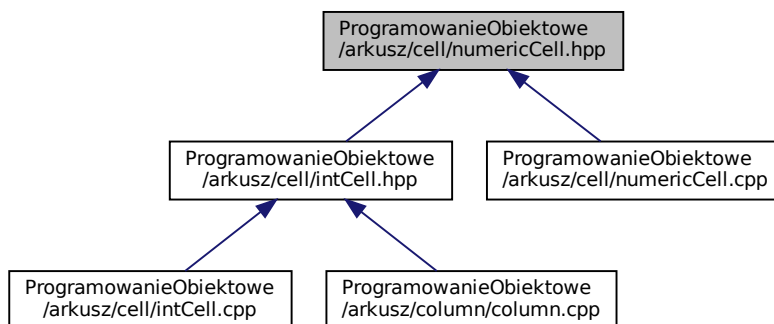
5.6 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/numericCell.hpp

```
#include "cell.hpp"
```

Wykres zależności załączania dla numericCell.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

- class [NumericCell](#)

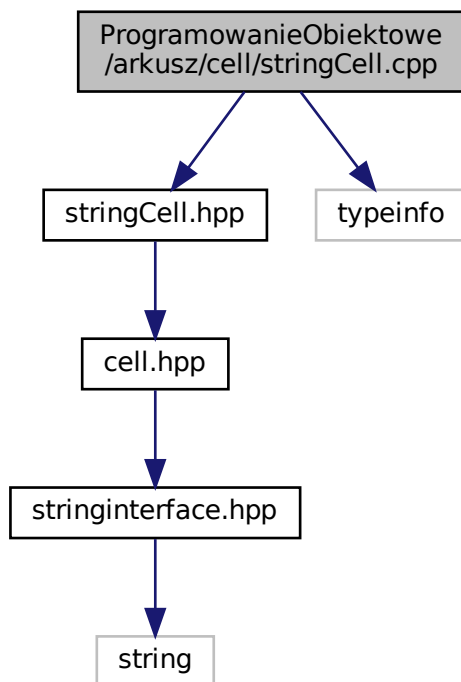
[NumericCell](#) komórki liczbowe Ogólna klasa abstrakcyjna komórek liczbowych które mogą posługiwać się wartościami rzeczywistymi.

5.7 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/stringCell.cpp

```
#include "stringCell.hpp"
```

```
#include <typeinfo>
```

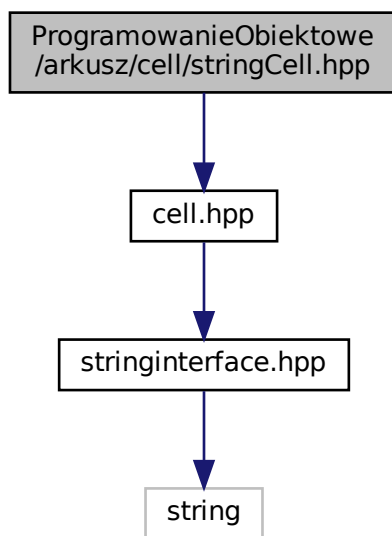
Wykres zależności załączania dla stringCell.cpp:



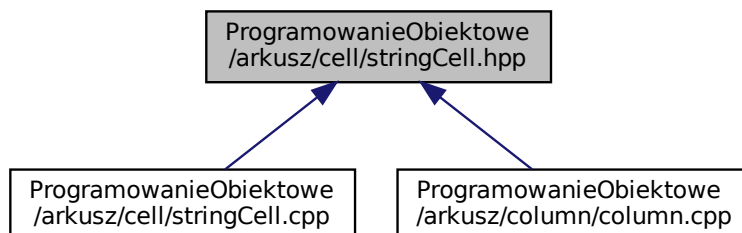
5.8 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/stringCell.hpp

```
#include "cell.hpp"
```

Wykres zależności załączania dla stringCell.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

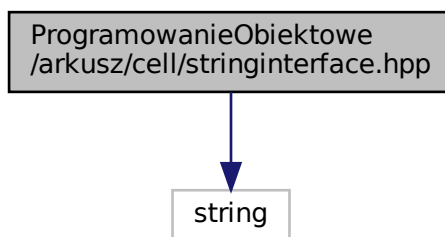
- class [StringCell](#)
StringCell Komórka tekstowa Klasa komórki przyjmującej wartości tekstowe.

5.9 Dokumentacja pliku

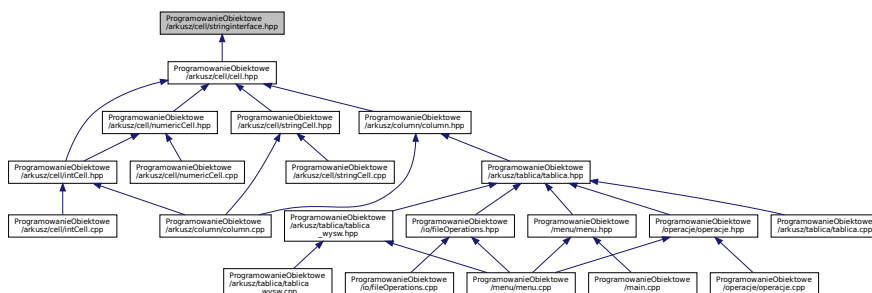
ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp

```
#include <string>
```

Wykres zależności załączania dla stringinterface.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

- class `StringInterface`
Intejejs elementów przyjmujących/zwracających elementy string.

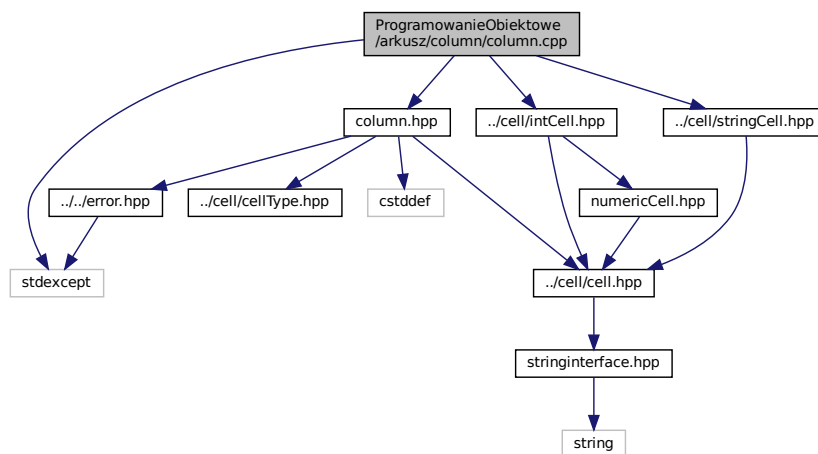
5.10 Dokumentacja pliku

ProgramowanieObiektowe/arkusz/column/column.cpp

```
#include "column.hpp"
#include "../cell/intCell.hpp"
#include "../cell/stringCell.hpp"
```

```
#include <stdexcept>
```

Wykres zależności załączania dla column.cpp:

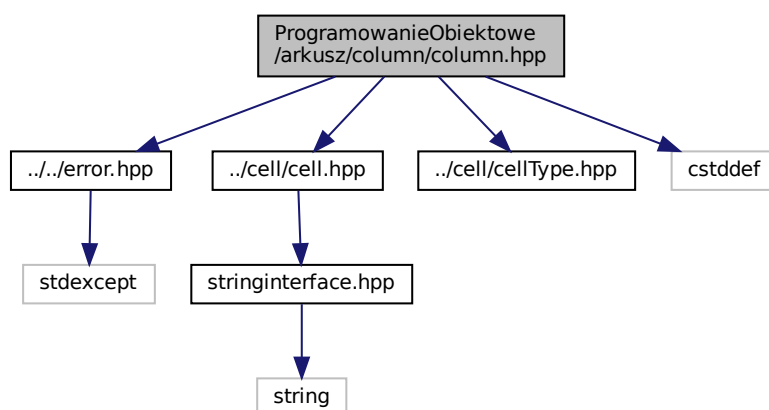


5.11 Dokumentacja pliku

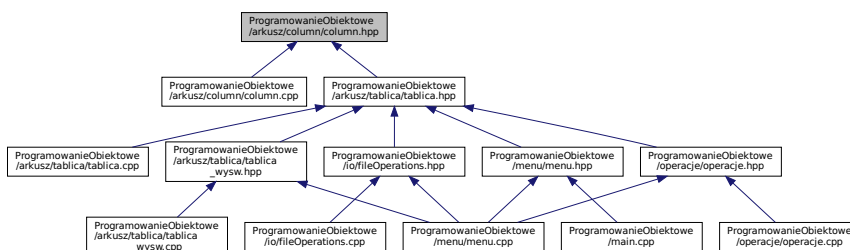
ProgramowanieObiektowe/arkusz/column/column.hpp

```
#include "../error.hpp"
#include "../cell/cell.hpp"
#include "../cell/cellType.hpp"
#include <cstddef>
```

Wykres zależności załączania dla column.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

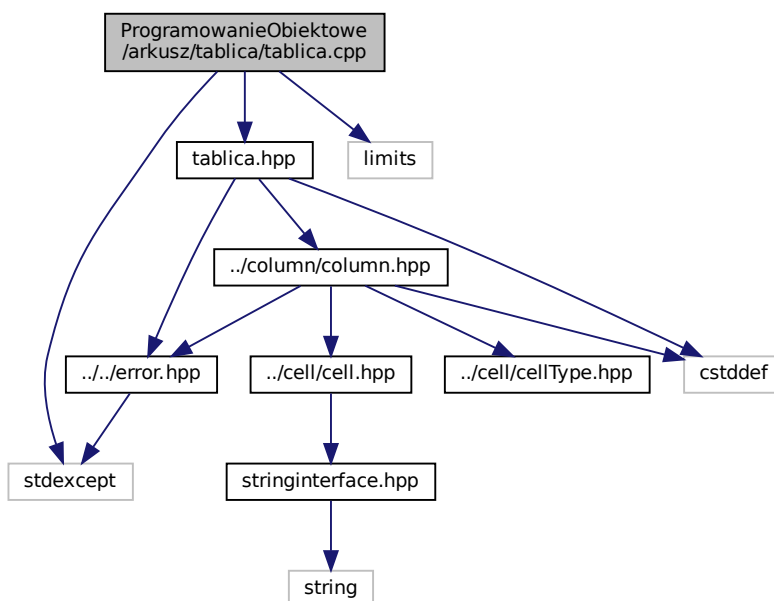
- class [Column](#)

Klasa określająca kolumnę. Klasa kolumna obsługująca podstawowe funkcjonalności kolekcji posiada wskaźnik początku i końca zakresu celem wykorzystania w funkcjach z biblioteki `algorithm` czy pętlach zakresowych.

5.12 Dokumentacja pliku ProgramowanieObiektowe/arkusz/tablica/tablica.cpp

```
#include "tablica.hpp"
#include <limits>
#include <stdexcept>
```

Wykres zależności załączania dla tablica.cpp:

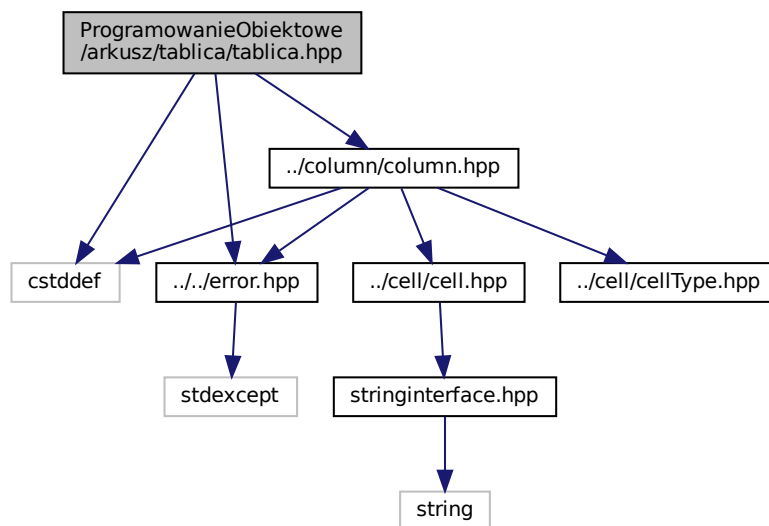


5.13 Dokumentacja pliku

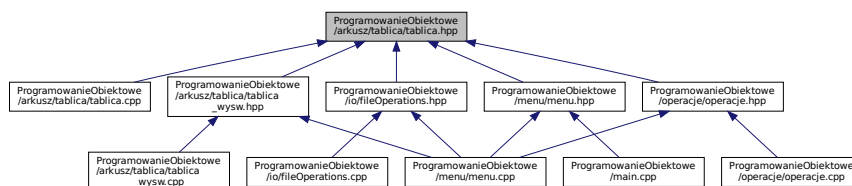
ProgramowanieObiektowe/arkusz/tablica/tablica.hpp

```
#include <cstddef>
#include "../error.hpp"
#include "../column/column.hpp"
```

Wykres zależności załączania dla tablica.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

- class [Sheet](#)

Klasa opisująca Arkusz Klasa Arkusz przechowywująca tablicę kolumn i jej rozmiar.

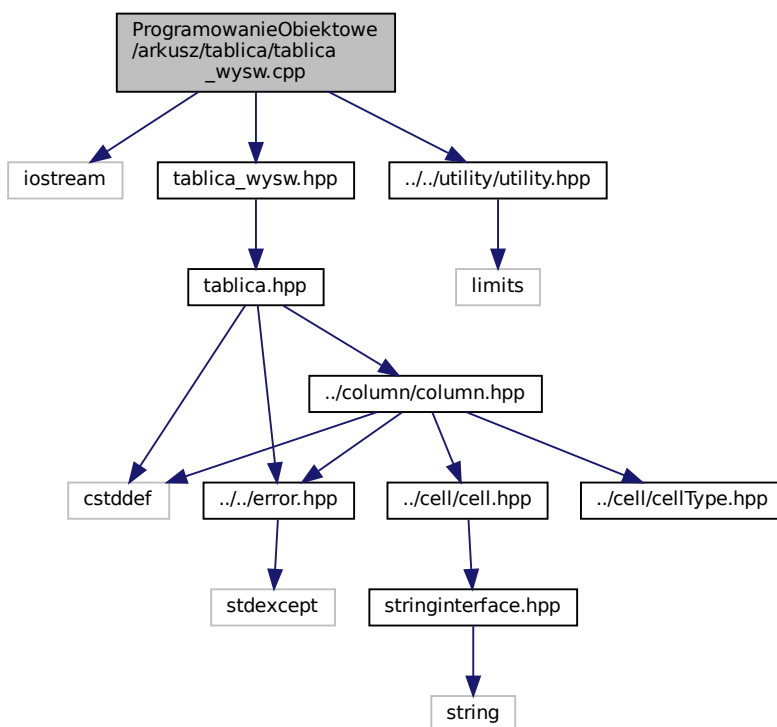
5.14 Dokumentacja pliku

ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.cpp

```
#include <iostream>
#include "tablica_wysw.hpp"
```

```
#include "../utility/utility.hpp"
```

Wykres zależności załączania dla tablica_wysw.cpp:



Funkcje

- void [DisplaySheet](#) ([Sheet](#) sheet)

Metoda od wyświetlania arkusza Metoda wyświetla wszystkie elementy znajdujące się w arkuszu.

5.14.1 Dokumentacja funkcji

5.14.1.1 DisplaySheet()

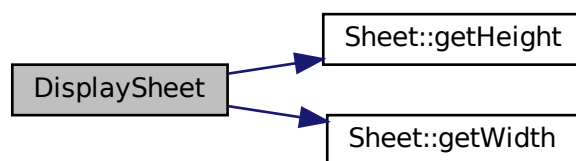
```
void DisplaySheet (
    Sheet sheet )
```

Metoda od wyświetlania arkusza Metoda wyświetla wszystkie elementy znajdujące się w arkuszu.

Parametry

in	sheet	Arkusz przeznaczony do wyświetlenia
----	-------	-------------------------------------

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

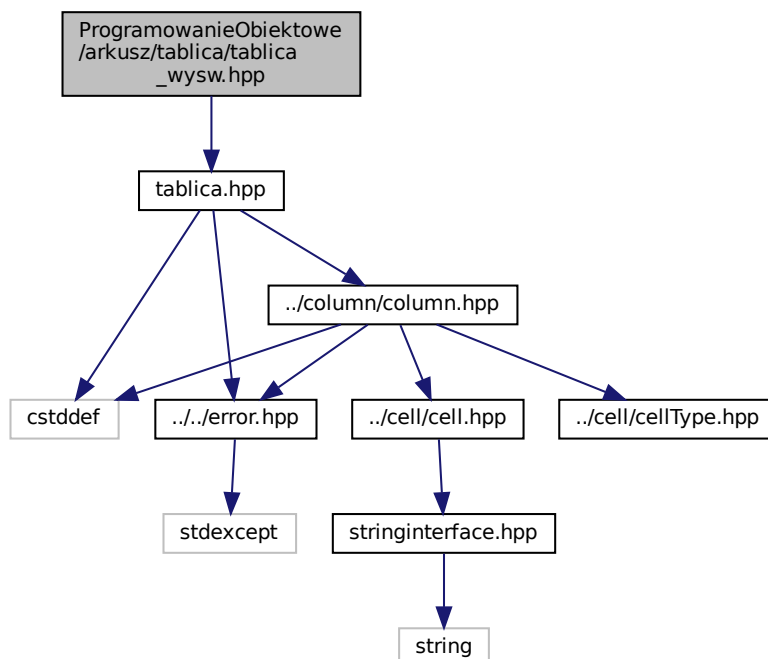


5.15 Dokumentacja pliku

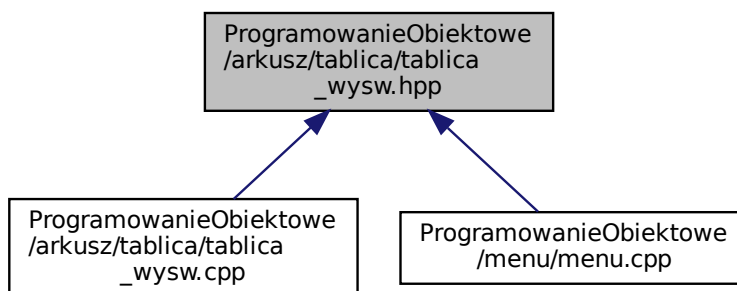
ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.hpp

```
#include "tablica.hpp"
```


Wykres zależności załączania dla tablica_wysw.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Funkcje

- void [DisplaySheet](#) ([Sheet](#) sheet)

Metoda od wyświetlania arkusza Metoda wyświetla wszystkie elementy znajdujące się w arkuszu.

5.15.1 Dokumentacja funkcji

5.15.1.1 DisplaySheet()

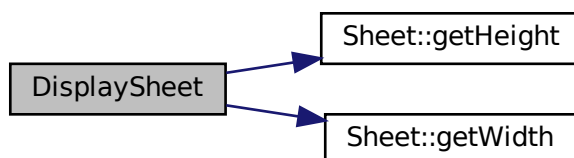
```
void DisplaySheet (
    Sheet sheet )
```

Metoda od wyświetlania arkusza Metoda wyświetla wszystkie elementy znajdujące się w arkuszu.

Parametry

in	sheet	Arkusz przeznaczony do wyświetlenia
----	-------	-------------------------------------

Oto graf wywołań dla tej funkcji:



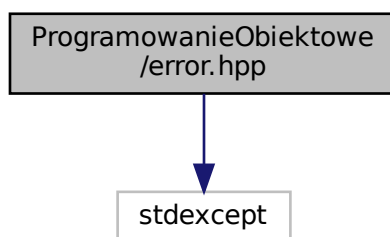
Oto graf wywoływań tej funkcji:



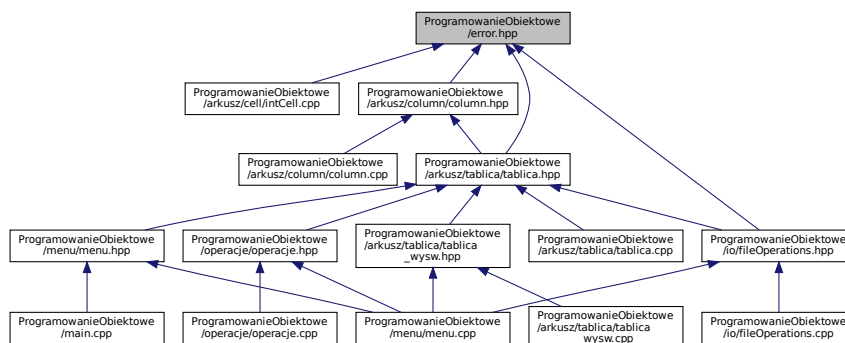
5.16 Dokumentacja pliku ProgramowanieObiektowe/error.hpp

```
#include <stdexcept>
```

Wykres zależności załączania dla error.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

- struct [BadFileException](#)

Wyjątek wyrzucany w przypadku braku dostępu. Wyjątek jest wyrzucany gdy użytkownik nie ma dostępu do pliku lub nie istnieje (odczyt).

- struct [NotNumericValue](#)

Wyjątek wyrzucany w przypadku wprowadzenia wartości nie numerycznej do komórki. Wyjątek powinien być wyrzucany gdy wprowadzony element nie może być przetworzony na wartość liczbową.

Wyliczenia

- enum class [Wyjatk](#) : unsigned int {
BRAK = 0 , **TABLICA_SIZE** = 1 , **TABLICA_ZAKR** = 2 , **PLIK_ACCESS** = 10 ,
PLIK_FORMAT = 11 , **PLIK_ROZMIAR** = 12 }

Wyjątki występujące w programie Typ wyliczeniowy który zawiera wszystkie występujące wyjątki.

5.16.1 Dokumentacja typów wyliczanych

5.16.1.1 Wyjątki

```
enum Wyjatki : unsigned int [strong]
```

Wyjątki występujące w programie Typ wycieniowy który zawiera wszystkie występujące wyjątki.

Wartości wycień

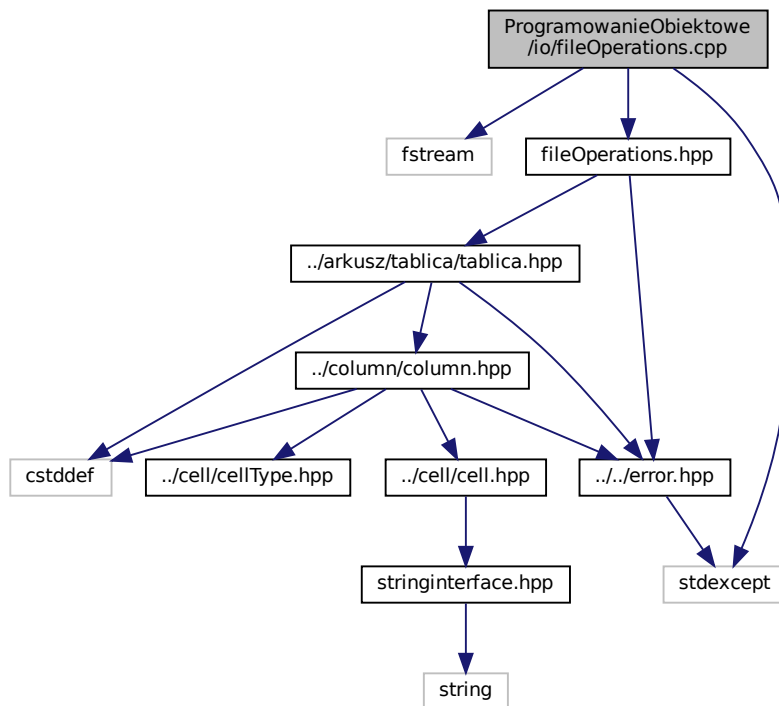
TABLICA_SIZE	Brak błędów.
TABLICA_ZAKR	Próba dostępu do elementu poza zakresem tablicy.
PLIK_ACCESS	Próba utworzenia tablicy o niepoprawnym rozmiarze.
PLIK_FORMAT	Niepoprawna nazwa lub brak dostępu do pliku.
PLIK_ROZMIAR	Niepoprawny format wczytywanego pliku.

5.17 Dokumentacja pliku

ProgramowanieObiektowe/io/fileOperations.cpp

```
#include <fstream>
#include <stdexcept>
#include "fileOperations.hpp"
```

Wykres zależności załączania dla fileOperations.cpp:



Funkcje

- void `saveFile` (`Sheet` sheet, `std::string` fileName)
Funkcja zapisu do pliku.
- void `loadFile` (`Sheet` *sheet, `std::string` fileName)
Funkcja wczytywania tablicy z pliku.

5.17.1 Dokumentacja funkcji

5.17.1.1 loadFile()

```
void loadFile (
    Sheet * sheet,
    std::string fileName )
```

Funkcja wczytywania tablicy z pliku.

Funkcja wykonuje wczytanie arkusza z wybranego pliku.

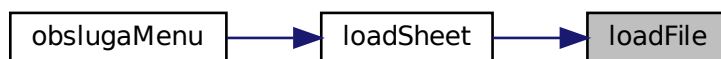
Parametry

in, out	<i>sheet</i>	Arkusz do nadpisania wczytywaną tablicą
in	<i>fileName</i>	Nazwa wczytywanego pliku

Wyjątki

<code>BadFileException</code>	W przypadku braku pliku lub braku dostępu
-------------------------------	---

Oto graf wywołań tej funkcji:



5.17.1.2 saveFile()

```
void saveFile (
    Sheet sheet,
    std::string fileName = "Arkusz.csv" )
```

Funkcja zapisu do pliku.

Funkcja wykonuje zapis do wybranego przez nas pliku

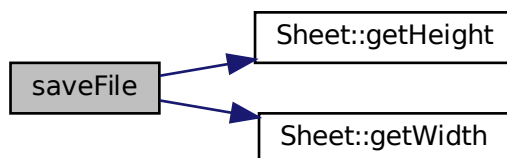
Parametry

in	<i>sheet</i>	Arkusz przeznaczony do zapisu
in	<i>fileName</i>	Nazwa zapisywanego pliku

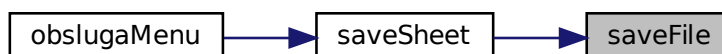
Wyjątki

<i>BadFileException</i>	W przypadku braku dostępu do zapisu
---	-------------------------------------

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

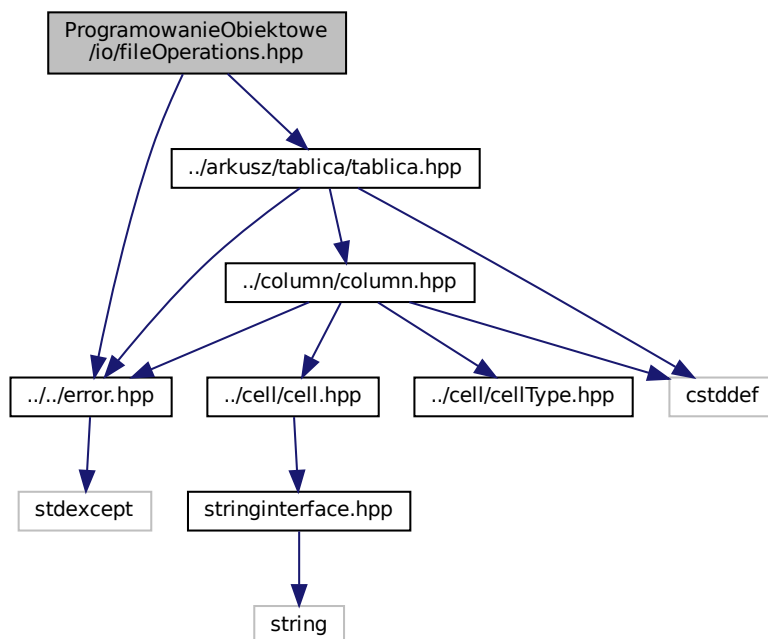


5.18 Dokumentacja pliku

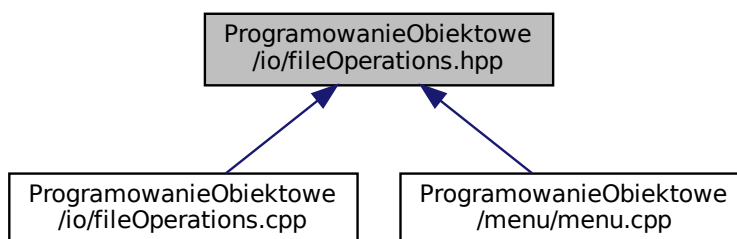
ProgramowanieObiektowe/io/fileOperations.hpp

```
#include "../arkusz/tablica/tablica.hpp"
#include "../error.hpp"
```

Wykres zależności załączania dla fileOperations.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Funkcje

- void [saveFile](#) ([Sheet](#) sheet, std::string fileName="Arkusz.csv")
Funkcja zapisu do pliku.
- void [loadFile](#) ([Sheet](#) *sheet, std::string fileName)
Funkcja wczytywania tablicy z pliku.

5.18.1 Dokumentacja funkcji

5.18.1.1 loadFile()

```
void loadFile (
    Sheet * sheet,
    std::string fileName )
```

Funkcja wczytywania tablicy z pliku.

Funkcja wykonuje wczytanie arkusza z wybranego pliku.

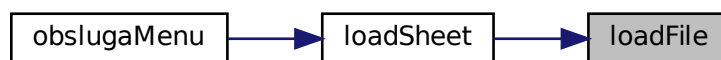
Parametry

in, out	<i>sheet</i>	Arkusz do nadpisania wczytywaną tablicą
in	<i>fileName</i>	Nazwa wczytywanego pliku

Wyjątki

<i>BadFileException</i>	W przypadku braku pliku lub braku dostępu
-------------------------	---

Oto graf wywołań tej funkcji:



5.18.1.2 saveFile()

```
void saveFile (
    Sheet sheet,
    std::string fileName = "Arkusz.csv" )
```

Funkcja zapisu do pliku.

Funkcja wykonuje zapis do wybranego przez nas pliku

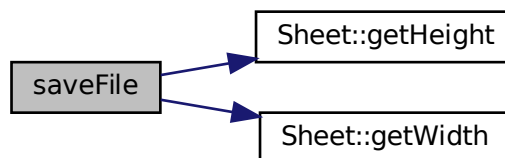
Parametry

in	<i>sheet</i>	Arkusz przeznaczony do zapisu
in	<i>fileName</i>	Nazwa zapisywanego pliku

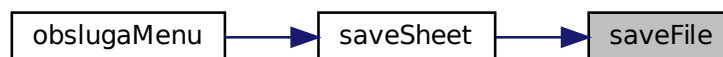
Wyjątki

<i>BadFileException</i>	W przypadku braku dostępu do zapisu
-------------------------	-------------------------------------

Oto graf wywołań dla tej funkcji:



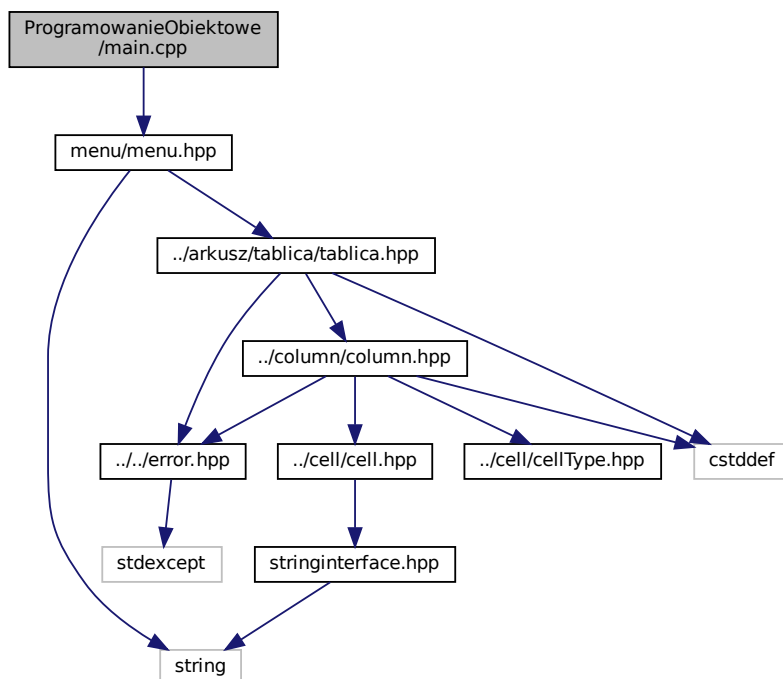
Oto graf wywoływań tej funkcji:



5.19 Dokumentacja pliku ProgramowanieObiektowe/main.cpp

```
#include "menu/menu.hpp"
```

Wykres zależności załączania dla main.cpp:



Funkcje

- `int main ()`

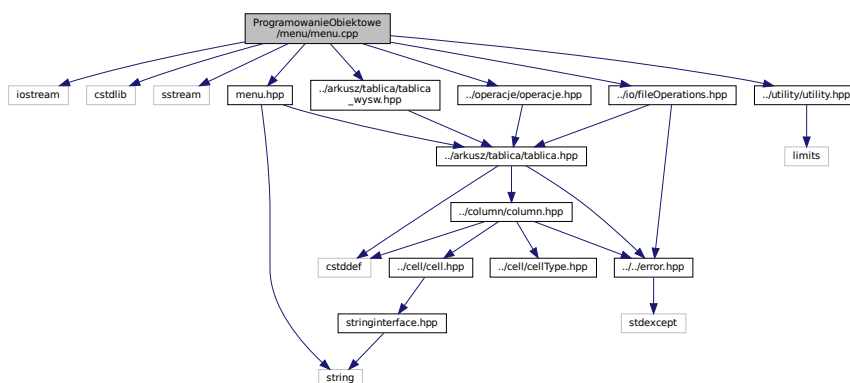
5.20 Dokumentacja pliku ProgramowanieObiektowe/menu/menu.cpp

```

#include <iostream>
#include <cstdlib>
#include <sstream>
#include "menu.hpp"
#include "../io/fileOperations.hpp"
#include "../arkusz/tablica/tablica_wysw.hpp"
#include "../utility/utility.hpp"
#include "../operacje/operacje.hpp"

```

Wykres zależności załączania dla menu.cpp:



Funkcje

- void **generujMenu** ()
Funkcja tworząca menu.
- void **obsługaMenu** ()
Funkcja kontrolująca działanie programu.
- void **loadSheet** (Sheet *arkusz)
Funkcja wczytywania arkusza Funkcja menu od wczytywania arkusza, ma za zadanie opakowanie funkcji IO wczytajPlik.
- void **saveSheet** (Sheet arkusz)
Funkcja menu od zapisu.
- **Sheet sheetCreator** ()
Funkcja tworząca nową tablicę.
- void **expandSheet** (Sheet *arkusz)
Funkcja modyfikująca rozmiar arkusza.
- void **inputValue** (Sheet *arkusz)
wprowadzWartosc modyfikuje wartość komórki Frontend od modyfikacji wartości który w zależności od typu komórki zmieni jej wartość
- void **sheetParameters** (Sheet arkusz)
Funkcja menu od wyboru względem czego wyznacza parametry.
- string **rowParameters** (Sheet arkusz, int wiersz)
Funkcja od wyznaczania parametrów wiersza arkusza.
- string **columnParameters** (Sheet arkusz, int kolumna)
Funkcja od wyznaczania parametrów kolumny arkusza.
- void **changeType** (Sheet *arkusz)
Funkcja od zmiany typu kolumny Interfejs od metody zmieniającej typ kolumny arkusza.
- void **sort** (Sheet *arkusz)
Funkcja interfejsu od sortowania kolumny Metoda zawiera interfejs do sortowania kolumny z arkusza.

5.20.1 Dokumentacja funkcji

5.20.1.1 changeType()

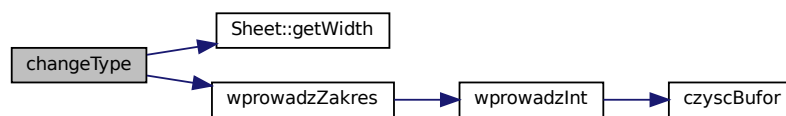
```
void changeType (
    Sheet * arkusz )
```

Funkcja od zmiany typu kolumny Interfejs od metody zmieniającej typ kolumny arkusza.

Parametry

in	arkusz	Arkusz którego kolumna zostaje zmieniona
----	--------	--

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.20.1.2 columnParameters()

```
string columnParameters (
    Sheet arkusz,
    int kolumna )
```

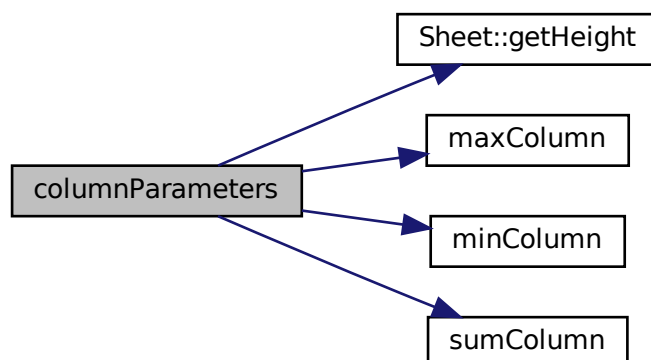
Funkcja od wyznaczania parametrów kolumny arkusza.

Funkcja zwraca w postaci tekstowej wszystkie parametry wybranej kolumny

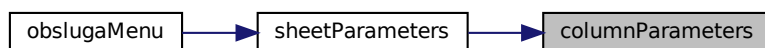
Parametry

in	arkusz	Przekazywany arkusz do wykorzystania w funkcji obliczania parametrów
in	kolumna	Kolumna względem której zostaną obliczone parametry

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.20.1.3 expandSheet()

```
void expandSheet (
    Sheet * arkusz )
```

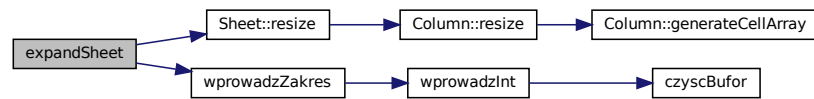
Funkcja modyfikująca rozmiar arkusza.

Interfejs umożliwiający modyfikację rozmiaru istniejącego arkusza.

Parametry

in, out	arkusz	Arkusz przeznaczony do modyfikacji rozmiaru
---------	--------	---

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.20.1.4 generujMenu()

```
void generujMenu ( )
```

Funkcja tworząca menu.

Funkcja od tworzenia listy dostępnych pozycji menu. Oto graf wywoływań tej funkcji:



5.20.1.5 inputValue()

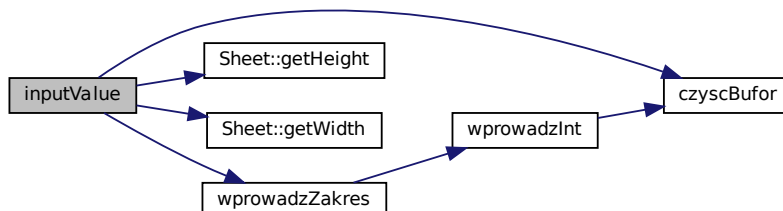
```
void inputValue (
    Sheet * arkusz )
```

wprowadzWartosc modyfikuje wartość komórki Frontend od modyfikacji wartości który w zależności od typu komórki zmieni jej wartość

Parametry

<i>arkusz</i>	Arkusz którego element będzie modyfikowany
---------------	--

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.20.1.6 loadSheet()

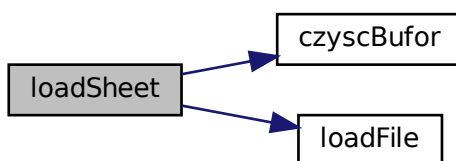
```
void loadSheet (
    Sheet * arkusz )
```

Funkcja wczytywania arkusza Funkcja menu od wczytywania arkusza, ma za zadanie opakowanie funkcji IO wczytajPlik.

Parametry

<i>in, out</i>	<i>arkusz</i>	Arkusz do którego mogą być wczytane elementy
----------------	---------------	--

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



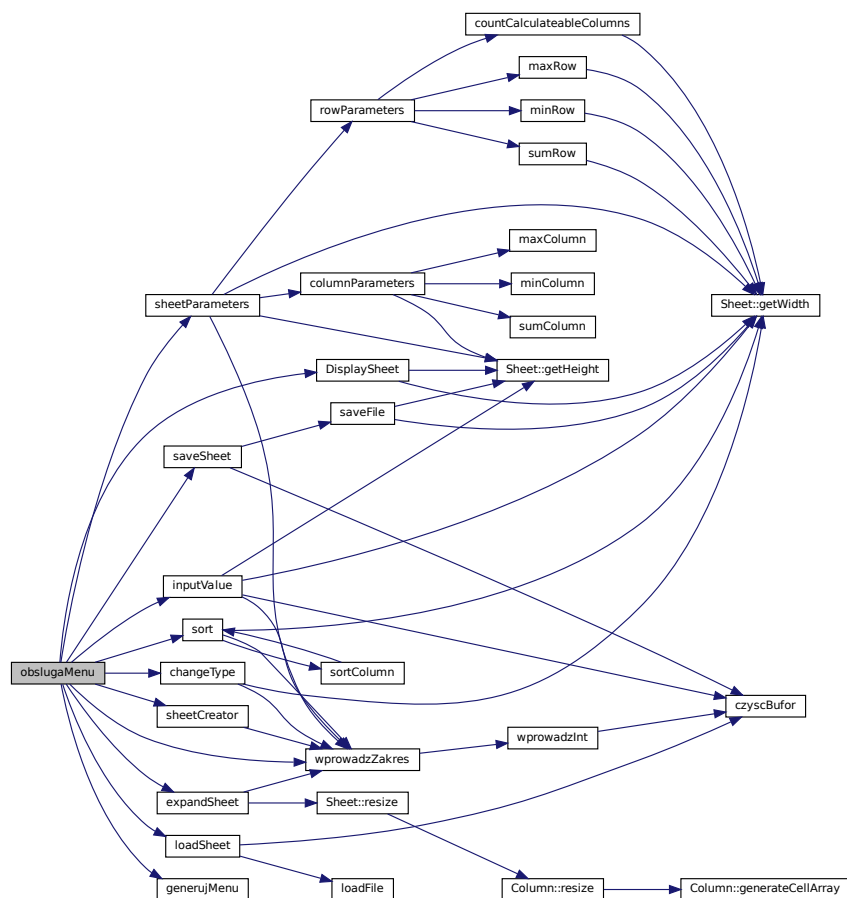
5.20.1.7 obsługaMenu()

```
void obsługaMenu ( )
```

Funkcja kontrolująca działanie programu.

Funkcja zajmująca się obsługą menu programu zarządza tym co będzie wywoływane. Oto graf wywołań dla tej

funkcji:



5.20.1.8 rowParameters()

```
string rowParameters (
    Sheet arkusz,
    int wiersz )
```

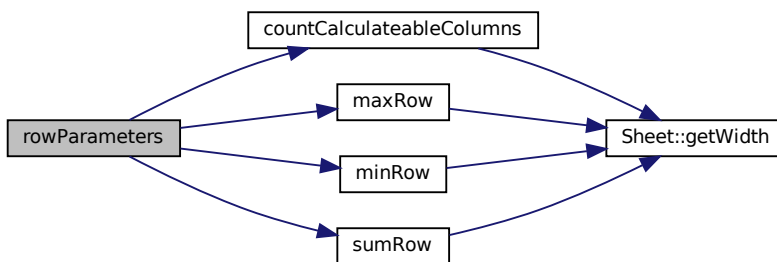
Funkcja od wyznaczania parametrów wiersza arkusza.

Funkcja zwraca w postaci tekstowej wszystkie parametry wybranego wiersza

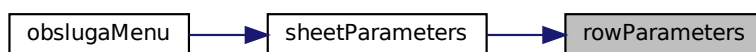
Parametry

in	arkusz	Przekazywany arkusz do wykorzystania w funkcji obliczania parametrów
in	wiersz	Wiersz względem której zostaną obliczone parametry

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.20.1.9 saveSheet()

```
void saveSheet (
    Sheet arkusz )
```

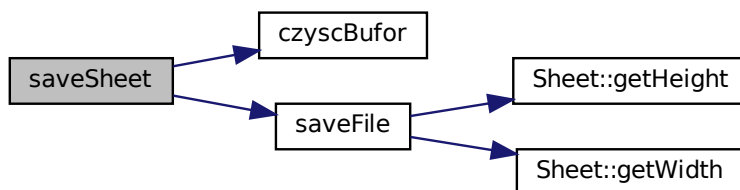
Funkcja menu od zapisu.

Funkcja menu od zapisu która ma za zadanie przetworzenie i opakowanie funkcji IO zapisPliku

Parametry

in	arkusz	Przekazywany arkusz do wykorzystania w funkcji zapisującej do pliku
----	--------	---

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.20.1.10 sheetCreator()

```
Sheet sheetCreator ( )
```

Funkcja tworząca nową tablicę.

Funkcja zawierająca interfejs umożliwiający tworzenie nowego Arkusza z tablicą dwuwymiarową.

Zwraca

Nowy Arkusz do wykorzystywania w programie

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



5.20.1.11 sheetParameters()

```
void sheetParameters (
    Sheet arkusz )
```

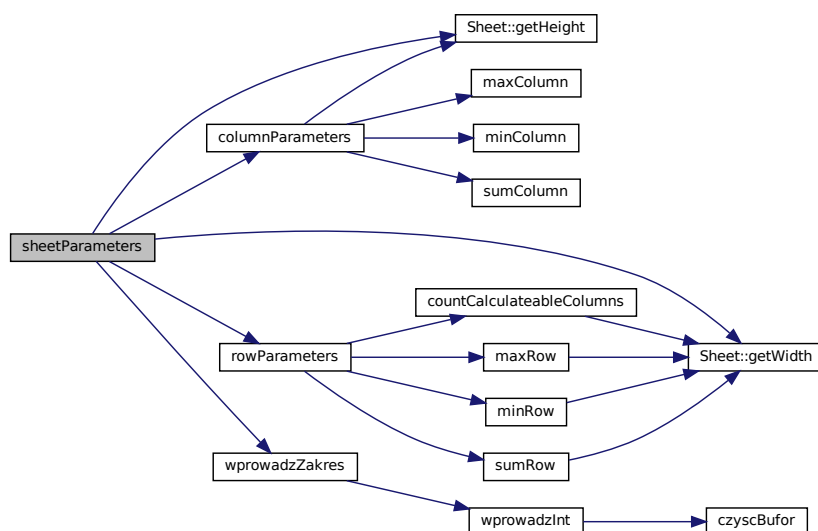
Funkcja menu od wyboru względem czego wyznacza parametry.

Funkcja menu od wyboru atrybutu tablicy (kolumny lub wiersza) która ma za wyświetlenie parametrów wybranego atrybutu.

Parametry

in	arkusz	Przekazywany arkusz do wykorzystania w funkcji wyboru parametrów
----	--------	--

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.20.1.12 sort()

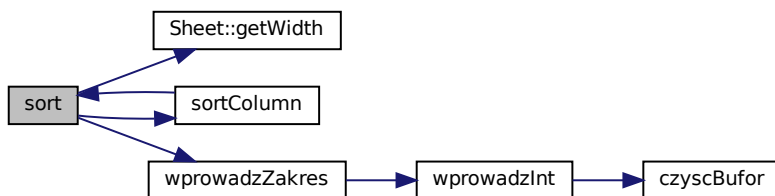
```
void sort (
    Sheet * arkusz )
```

Funkcja interfejsu od sortowania kolumny Metoda zawiera interfejs do sortowania kolumny z arkusza.

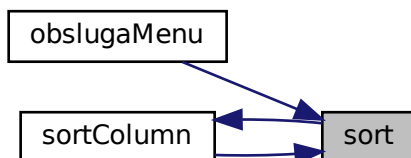
Parametry

<code>in, out</code>	<i>Arkusz</i>	którego kolumna będzie sortowana
----------------------	---------------	----------------------------------

Oto graf wywołań dla tej funkcji:



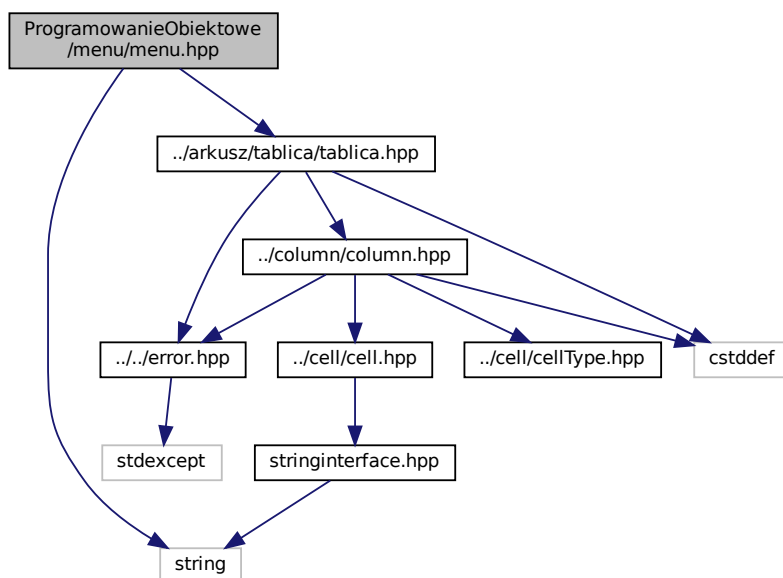
Oto graf wywoływań tej funkcji:



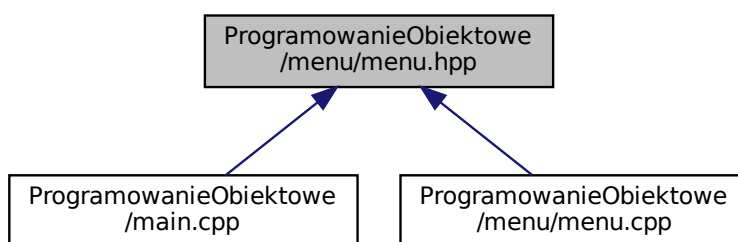
5.21 Dokumentacja pliku ProgramowanieObiektowe/menu/menu.hpp

```
#include <string>
#include "../arkusz/tablica/tablica.hpp"
```

Wykres zależności załączania dla menu.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Funkcje

- void [obsługaMenu](#) ()
Funkcja kontrolująca działanie programu.
- void [generujMenu](#) ()
Funkcja tworząca menu.
- [Sheet](#) [sheetCreator](#) ()

- *Funkcja tworząca nową tablicę.*
• void `expandSheet` (`Sheet` *arkusz)
- *Funkcja modyfikująca rozmiar arkusza.*
• void `loadSheet` (`Sheet` *arkusz)
Funkcja wczytywania arkusza Funkcja menu od wczytywania arkusza, ma za zadanie opakowanie funkcji IO wczytajPlik.
- void `saveSheet` (`Sheet` arkusz)
Funkcja menu od zapisu.
- void `sheetParameters` (`Sheet` arkusz)
Funkcja menu od wyboru względem czego wyznacza parametry.
- std::string `rowParameters` (`Sheet` arkusz, int wiersz)
Funkcja od wyznaczania parametrów wiersza arkusza.
- std::string `columnParameters` (`Sheet` arkusz, int kolumna)
Funkcja od wyznaczania parametrów kolumny arkusza.
- void `inputValue` (`Sheet` *arkusz)
wprowadzWartosc modyfikuje wartość komórki Frontend od modyfikacji wartości który w zależności od typu komórki zmieni jej wartość
- void `changeType` (`Sheet` *arkusz)
Funkcja od zmiany typu kolumny Interfejs od metody zmieniającej typ kolumny arkusza.
- void `sort` (`Sheet` *arkusz)
Funkcja interfejsu od sortowania kolumny Metoda zawiera interfejs do sortowania kolumny z arkusza.

5.21.1 Dokumentacja funkcji

5.21.1.1 changeType()

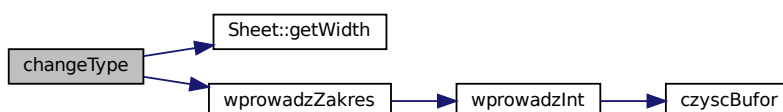
```
void changeType (
    Sheet * arkusz )
```

Funkcja od zmiany typu kolumny Interfejs od metody zmieniającej typ kolumny arkusza.

Parametry

in	arkusz	Arkusz którego kolumna zostaje zmieniona
----	--------	--

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



5.21.1.2 columnParameters()

```
std::string columnParameters (
    Sheet arkusz,
    int kolumna )
```

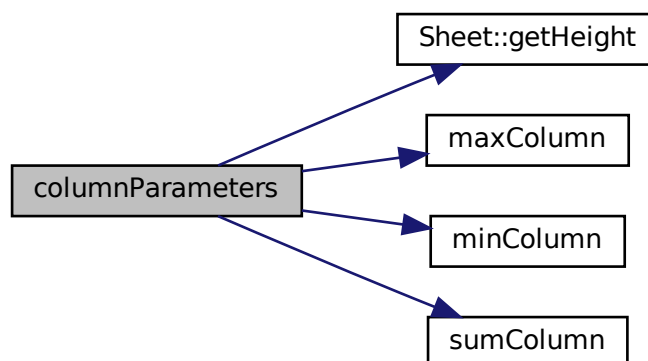
Funkcja od wyznaczania parametrów kolumny arkusza.

Funkcja zwraca w postaci tekstowej wszystkie parametry wybranej kolumny

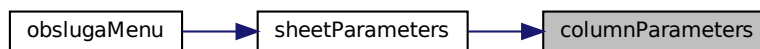
Parametry

in	<i>arkusz</i>	Przekazywany arkusz do wykorzystania w funkcji obliczania parametrów
in	<i>kolumna</i>	Kolumna względem której zostaną obliczone parametry

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.21.1.3 expandSheet()

```
void expandSheet (
    Sheet * arkusz )
```

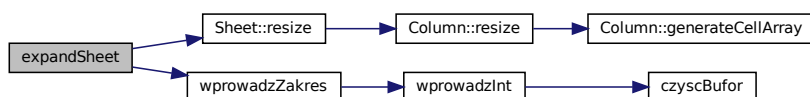
Funkcja modyfikująca rozmiar arkusza.

Interfejs umożliwiający modyfikację rozmiaru istniejącego arkusza.

Parametry

in, out	arkusz	Arkusz przeznaczony do modyfikacji rozmiaru
---------	--------	---

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.21.1.4 generujMenu()

```
void generujMenu ( )
```

Funkcja tworząca menu.

Funkcja od tworzenia listy dostępnych pozycji menu. Oto graf wywołań tej funkcji:



5.21.1.5 inputValue()

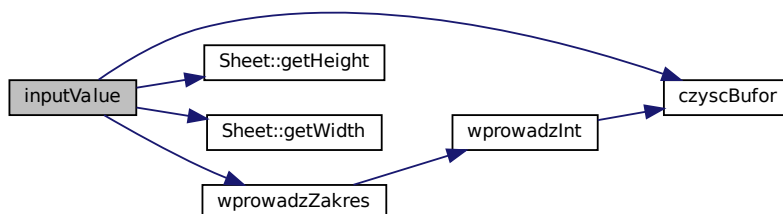
```
void inputValue (
    Sheet * arkusz )
```

wprowadzWartosc modyfikuje wartość komórki Frontend od modyfikacji wartości który w zależności od typu komórki zmieni jej wartość

Parametry

arkusz	Arkusz którego element będzie modyfikowany
--------	--

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.21.1.6 loadSheet()

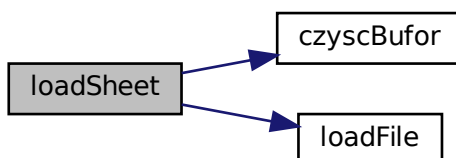
```
void loadSheet (
    Sheet * arkusz )
```

Funkcja wczytywania arkusza Funkcja menu od wczytywania arkusza, ma za zadanie opakowanie funkcji IO wczytajPlik.

Parametry

<code>in, out</code>	<code>arkusz</code>	Arkusz do którego mogą być wczytane elementy
----------------------	---------------------	--

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

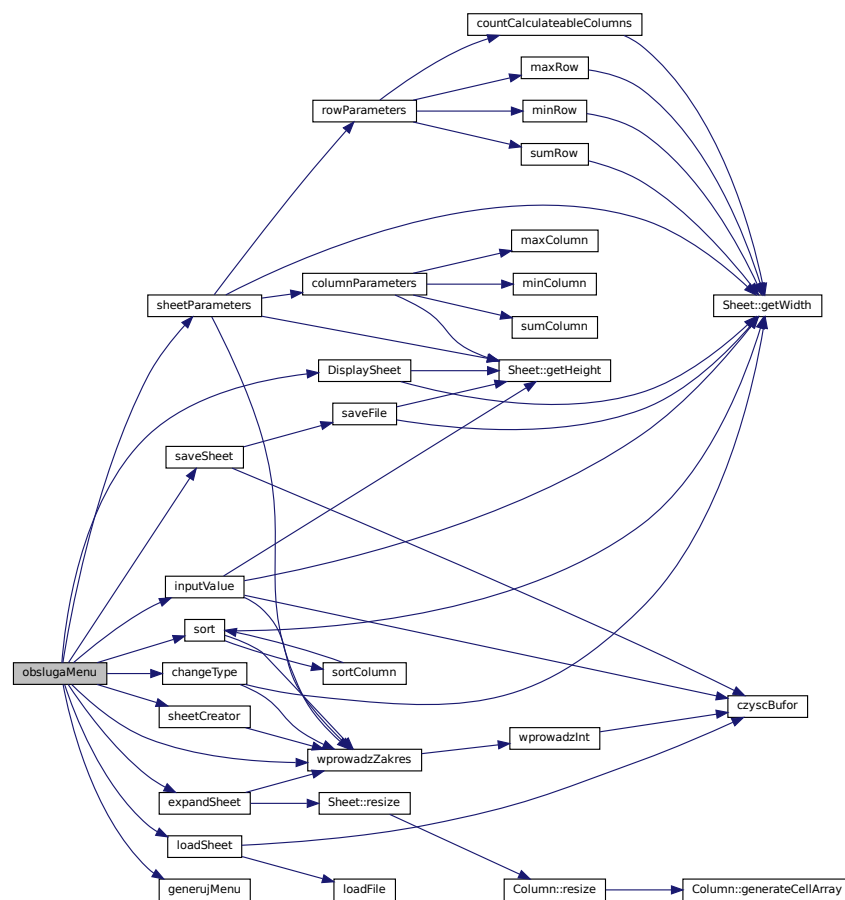


5.21.1.7 obsługaMenu()

```
void obsługaMenu ( )
```

Funkcja kontrolująca działanie programu.

Funkcja zajmująca się obsługą menu programu zarządza tym co będzie wywoływane Oto graf wywołań dla tej funkcji:



5.21.1.8 rowParameters()

```
std::string rowParameters (
    Sheet arkusz,
    int wiersz )
```

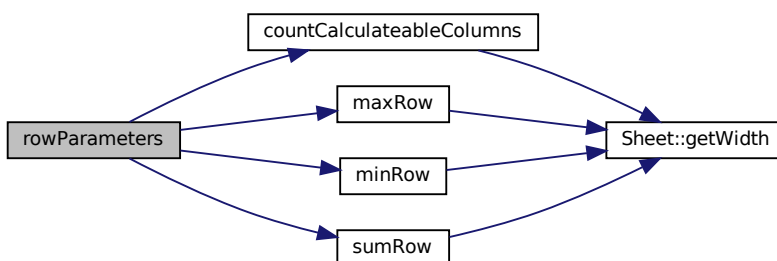
Funkcja od wyznaczania parametrów wiersza arkusza.

Funkcja zwraca w postaci tekstowej wszystkie parametry wybranego wiersza

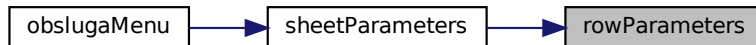
Parametry

in	arkusz	Przekazywany arkusz do wykorzystania w funkcji obliczania parametrów
in	wiersz	Wiersz względem której zostaną obliczone parametry

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.21.1.9 saveSheet()

```
void saveSheet (
    Sheet arkusz )
```

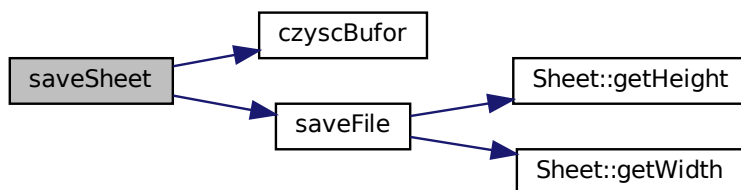
Funkcja menu od zapisu.

Funkcja menu od zapisu która ma za zadanie przetworzenie i opakowanie funkcji IO zapisPliku

Parametry

in	arkusz	Przekazywany arkusz do wykorzystania w funkcji zapisującej do pliku
----	--------	---

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.21.1.10 sheetCreator()

```
Sheet sheetCreator ( )
```

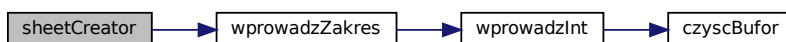
Funkcja tworząca nową tablicę.

Funkcja zawierająca interfejs umożliwiający tworzenie nowego Arkusza z tablicą dwuwymiarową.

Zwraca

Nowy Arkusz do wykorzystywania w programie

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



5.21.1.11 sheetParameters()

```
void sheetParameters (
    Sheet arkusz )
```

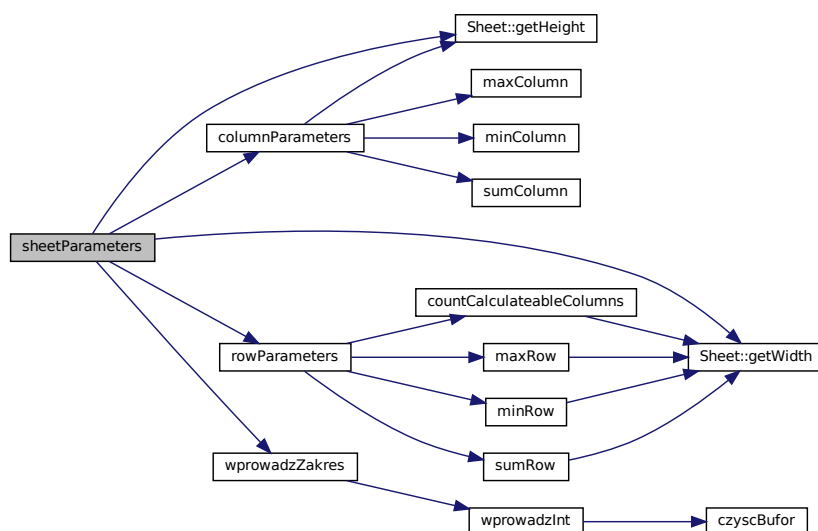
Funkcja menu od wyboru względem czego wyznacza parametry.

Funkcja menu od wyboru atrybutu tablicy (kolumny lub wiersza) która ma za wyświetlenie parametrów wybranego atrybutu.

Parametry

in	arkusz	Przekazywany arkusz do wykorzystania w funkcji wyboru parametrów
----	--------	--

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.21.1.12 sort()

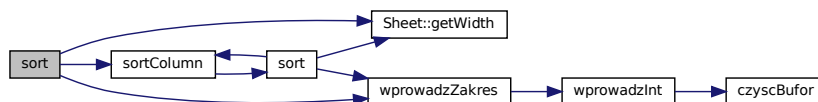
```
void sort (
    Sheet * arkusz )
```

Funkcja interfejsu od sortowania kolumny Metoda zawiera interfejs do sortowania kolumny z arkusza.

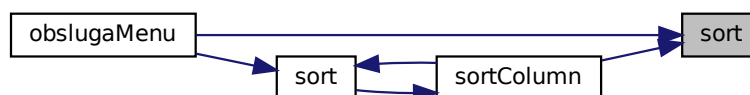
Parametry

in, out	Arkusz	którego kolumna będzie sortowana
---------	--------	----------------------------------

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.22 Dokumentacja pliku

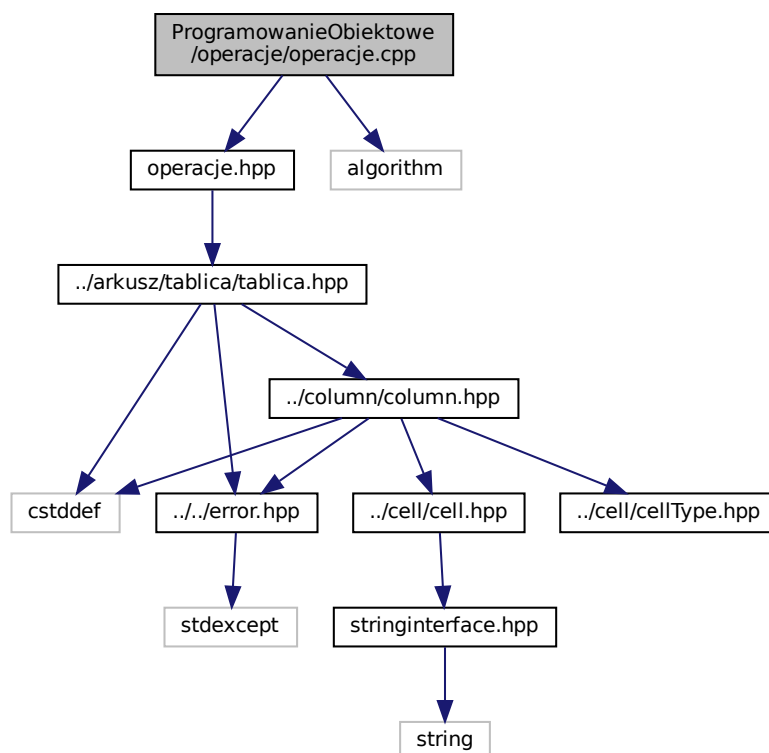
ProgramowanieObiektowe/operacje/operacje.cpp

```
#include "operacje.hpp"
```



```
#include <algorithm>
```

Wykres zależności załączania dla operacje.cpp:



Funkcje

- double **maxRow** (**Sheet** sheet, size_t row)
Funkcja szukania maksymalnej wartości wiersza.
- double **minRow** (**Sheet** sheet, size_t row)
Funkcja szukania minimalnej wartości wiersza.
- double **sumRow** (**Sheet** sheet, size_t row)
Funkcja licząca sumę elementów wiersza.
- double **maxColumn** (**Column** column)
Funkcja szukania maksymalnej wartości kolumny.
- double **minColumn** (**Column** column)
Funkcja szukania minimalnej wartości kolumny.
- double **sumColumn** (**Column** column)
Funkcja licząca sumę elementów kolumny.
- void **sortColumn** (**Column** *column, bool descending)
sortKolumna Sortuje komórki w kolumnie rosnąco lub malejąco Sortuje kolumnę w zależności od podanego parametru descending
- int **countCalculateableColumns** (**Sheet** sheet)
countCalculateableRow Liczy ilość kolumn obliczalnych Liczy ile jest kolumn które mogą być wykorzystywane przy obliczeniach

5.22.1 Dokumentacja funkcji

5.22.1.1 countCalculateableColumns()

```
int countCalculateableColumns (
    Sheet sheet )
```

countCalculateableRow Liczy ilość kolumn obliczalnych Liczy ile jest kolumn które mogą być wykorzystywane przy obliczeniach

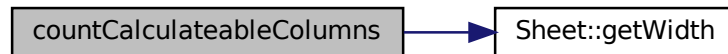
Parametry

<i>sheet</i>	Arkusz którego elementy będą liczone
--------------	--------------------------------------

Zwraca

liczba kolumn typów obliczalnych

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.22.1.2 maxColumn()

```
double maxColumn (
    Column column )
```

Funkcja szukania maksymalnej wartości kolumny.

Funkcja przeszukuje całą kolumnę celem określenia największej wprowadzonej wartości

Parametry

in	<i>column</i>	Kolumna względem którego obliczamy parametr
----	---------------	---

Zwraca

Zwraca wartość największą kolumny

Oto graf wywołań tej funkcji:

**5.22.1.3 maxRow()**

```
double maxRow (
    Sheet sheet,
    size_t row )
```

Funkcja szukania maksymalnej wartości wiersza.

Funkcja przeszukuje cały wiersz celem określenia największej wprowadzonej wartości

Parametry

in	<i>sheet</i>	Arkusz przeznaczony do obliczania parametru
in	<i>row</i>	Wiersz względem którego obliczamy parametr

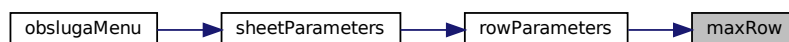
Zwraca

Zwraca wartość maksymalną wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.22.1.4 minColumn()

```
double minColumn (
    Column column )
```

Funkcja szukania minimalnej wartości kolumny.

Funkcja przeszukuje całą kolumnę celem określenia najmniejszej wprowadzonej wartości

Parametry

in	column	Kolumna względem którego obliczamy parametr
----	--------	---

Zwraca

Zwraca wartość najmniejszą kolumny

Oto graf wywoływań tej funkcji:



5.22.1.5 minRow()

```
double minRow (
    Sheet sheet,
    size_t row )
```

Funkcja szukania minimalnej wartości wiersza.

Funkcja przeszukuje cały wiersz celem określenia najmniejszej wprowadzonej wartości

Parametry

in	<i>sheet</i>	Arkusz przeznaczony do obliczania parametru
in	<i>row</i>	Wiersz względem którego obliczamy parametr

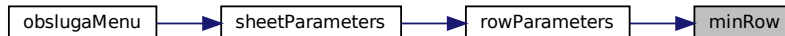
Zwraca

Zwraca wartość najmniejszą wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

**5.22.1.6 sortColumn()**

```

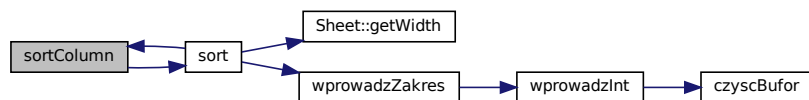
void sortColumn (
    Column * column,
    bool descending = false )
  
```

sortKolumna Sortuje komórki w kolumnie rosnąco lub malejąco Sortuje kolumnę w zależności od podanego parametru descending

Parametry

in, out	<i>column</i>	Kolumna przeznaczona do sortowania
in	<i>descending</i>	Definiuje czy kolumna będzie sortowana rosnąco lub malejąco

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.22.1.7 sumColumn()

```
double sumColumn (
    Column column )
```

Funkcja licząca sumę elementów kolumny.

Funkcja zwraca sumę całej kolumny

Parametry

in	column	Kolumna względem którego obliczamy parametr
----	--------	---

Zwraca

Zwraca sumę wszystkich elementów kolumny

Oto graf wywoływań tej funkcji:



5.22.1.8 sumRow()

```
double sumRow (
    Sheet sheet,
    size_t row )
```

Funkcja licząca sumę elementów wiersza.

Funkcja zwraca sumę całego wiersza

Parametry

in	sheet	Arkusz przeznaczony do obliczania parametru
in	row	Wiersz względem którego obliczamy parametr

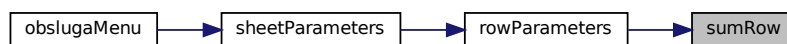
Zwraca

Zwraca sumę wszystkich elementów wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

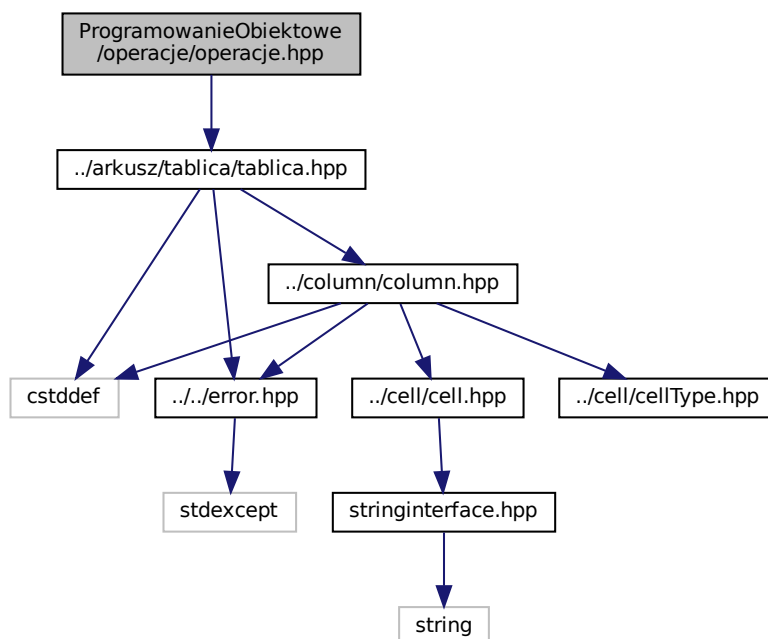


5.23 Dokumentacja pliku

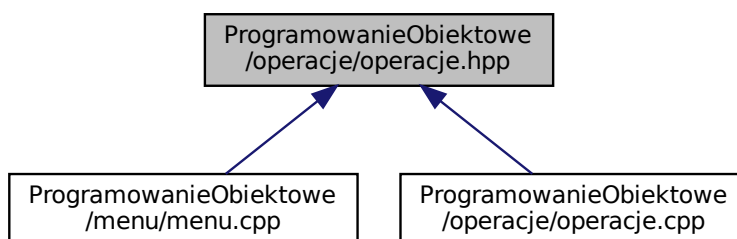
ProgramowanieObiektowe/operacje/operacje.hpp

```
#include "../arkusz/tablica/tablica.hpp"
```

Wykres zależności załączania dla operacje.hpp:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Funkcje

- double [maxRow](#) ([Sheet](#) sheet, size_t row)
Funkcja szukania maksymalnej wartości wiersza.
- double [minRow](#) ([Sheet](#) sheet, size_t row)
Funkcja szukania minimalnej wartości wiersza.
- double [sumRow](#) ([Sheet](#) sheet, size_t row)
Funkcja licząca sumę elementów wiersza.
- int [countCalculateableColumns](#) ([Sheet](#) sheet)

countCalculateableRow Liczy ilość kolumn obliczalnych Liczy ile jest kolumn które mogą być wykorzystywane przy obliczeniach

- double **maxColumn** (**Column** column)

Funkcja szukania maksymalnej wartości kolumny.

- double **minColumn** (**Column** column)

Funkcja szukania minimalnej wartości kolumny.

- double **sumColumn** (**Column** column)

Funkcja licząca sumę elementów kolumny.

- void **sortColumn** (**Column** *column, bool descending=false)

sortKolumna Sortuje komórki w kolumnie rosnąco lub malejąco Sortuje kolumnę w zależności od podanego parametru *descending*

5.23.1 Dokumentacja funkcji

5.23.1.1 countCalculateableColumns()

```
int countCalculateableColumns (
    Sheet sheet )
```

countCalculateableRow Liczy ilość kolumn obliczalnych Liczy ile jest kolumn które mogą być wykorzystywane przy obliczeniach

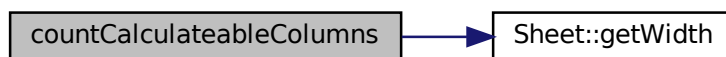
Parametry

<i>sheet</i>	Arkusz którego elementy będą liczone
--------------	--------------------------------------

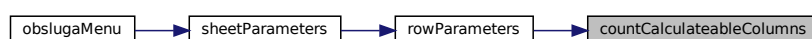
Zwraca

liczba kolumn typów obliczalnych

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.23.1.2 maxColumn()

```
double maxColumn (  
    Column column )
```

Funkcja szukania maksymalnej wartości kolumny.

Funkcja przeszukuje całą kolumnę celem określenia największej wprowadzonej wartości

Parametry

in	<i>column</i>	Kolumna względem którego obliczamy parametr
----	---------------	---

Zwraca

Zwraca wartość największą kolumny

Oto graf wywołań tej funkcji:



5.23.1.3 maxRow()

```
double maxRow (  
    Sheet sheet,  
    size_t row )
```

Funkcja szukania maksymalnej wartości wiersza.

Funkcja przeszukuje cały wiersz celem określenia największej wprowadzonej wartości

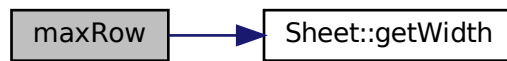
Parametry

in	<i>sheet</i>	Arkusz przeznaczony do obliczania parametru
in	<i>row</i>	Wiersz względem którego obliczamy parametr

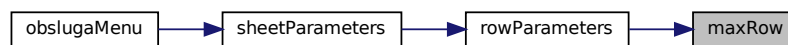
Zwraca

Zwraca wartość maksymalną wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.23.1.4 minColumn()

```
double minColumn (
    Column column )
```

Funkcja szukania minimalnej wartości kolumny.

Funkcja przeszukuje całą kolumnę celem określenia najmniejszej wprowadzonej wartości

Parametry

in	column	Kolumna względem którego obliczamy parametr
----	--------	---

Zwraca

Zwraca wartość najmniejszą kolumny

Oto graf wywoływań tej funkcji:



5.23.1.5 minRow()

```
double minRow (
    Sheet sheet,
    size_t row )
```

Funkcja szukania minimalnej wartości wiersza.

Funkcja przeszukuje cały wiersz celem określenia najmniejszej wprowadzonej wartości

Parametry

in	<i>sheet</i>	Arkusz przeznaczony do obliczania parametru
in	<i>row</i>	Wiersz względem którego obliczamy parametr

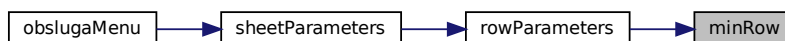
Zwraca

Zwraca wartość najmniejszą wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.23.1.6 sortColumn()

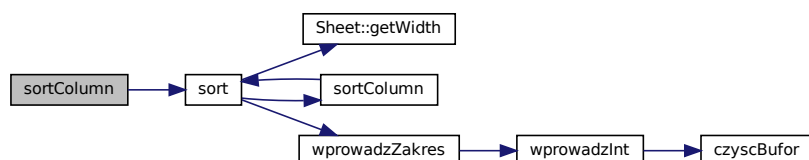
```
void sortColumn (
    Column * column,
    bool descending = false )
```

sortKolumna Sortuje komórki w kolumnie rosnąco lub malejąco Sortuje kolumnę w zależności od podanego parametru descending

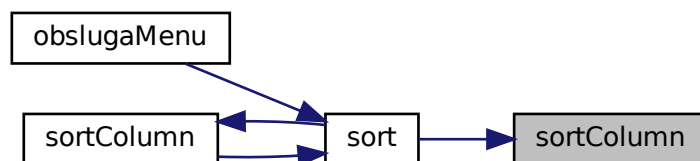
Parametry

in, out	<i>column</i>	Kolumna przeznaczona do sortowania
in	<i>descending</i>	Definiuje czy kolumna będzie sortowana rosnąco lub malejąco

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.23.1.7 sumColumn()

```
double sumColumn (
    Column column )
```

Funkcja licząca sumę elementów kolumny.

Funkcja zwraca sumę całej kolumny

Parametry

in	<i>column</i>	Kolumna względem którego obliczamy parametr
----	---------------	---

Zwraca

Zwraca sumę wszystkich elementów kolumny

Oto graf wywołań tej funkcji:



5.23.1.8 sumRow()

```
double sumRow (
    Sheet sheet,
    size_t row )
```

Funkcja licząca sumę elementów wiersza.

Funkcja zwraca sumę całego wiersza

Parametry

in	sheet	Arkusz przeznaczony do obliczania parametru
in	row	Wiersz względem którego obliczamy parametr

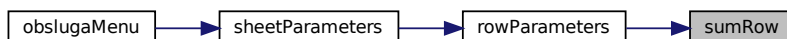
Zwraca

Zwraca sumę wszystkich elementów wiersza

Oto graf wywołań dla tej funkcji:



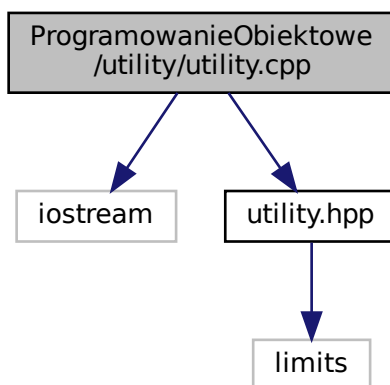
Oto graf wywołań tej funkcji:



5.24 Dokumentacja pliku ProgramowanieObiektowe/utility/utility.cpp

```
#include <iostream>
#include "utility.hpp"
```

Wykres zależności załączania dla utility.cpp:



Funkcje

- int `wprowadzInt()`
funkcja od wprowadzania wartości typu int
- int `wprowadzZakres(int min, int max)`
Wprowadź wartość z wyznaczonego zakresu w konsoli Funkcja służy do zwracania wartości wprowadzonej konsolowo, która mieści się wyznaczonym zakresie.
- void `czyszcBufor()`
Funkcja od czyszczenia buforu strumienia CIN.

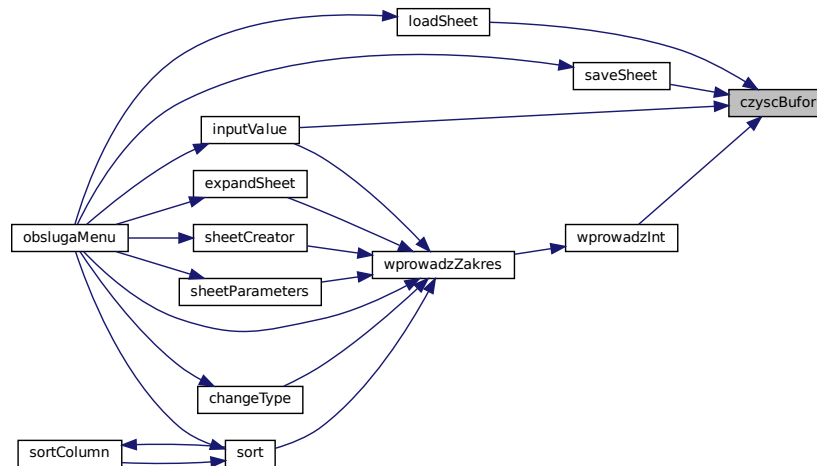
5.24.1 Dokumentacja funkcji

5.24.1.1 czyszcBufor()

```
void czyszcBufor ( )
```

Funkcja od czyszczenia buforu strumienia CIN.

Funkcja ma za zadanie wyczyścić bufor strumienia wejściowego CIN celem wprowadzenia np. string'a Oto graf wywołań tej funkcji:



5.24.1.2 wprowadzZakres()

```

int wprowadzZakres (
    int min = 1,
    int max = std::numeric_limits< int >::max() )
  
```

Wprowadź wartość z wyznaczonego zakresu w konsoli Funkcja służy do zwracania wartości wprowadzonej konsolowo, która mieści się wyznaczonym zakresie.

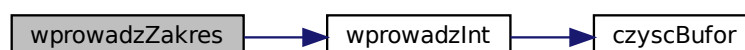
Parametry

in	<i>min</i>	Minimalna wartość jaką można wprowadzić
in	<i>max</i>	Maksymalna wartość jaką można wprowadzić

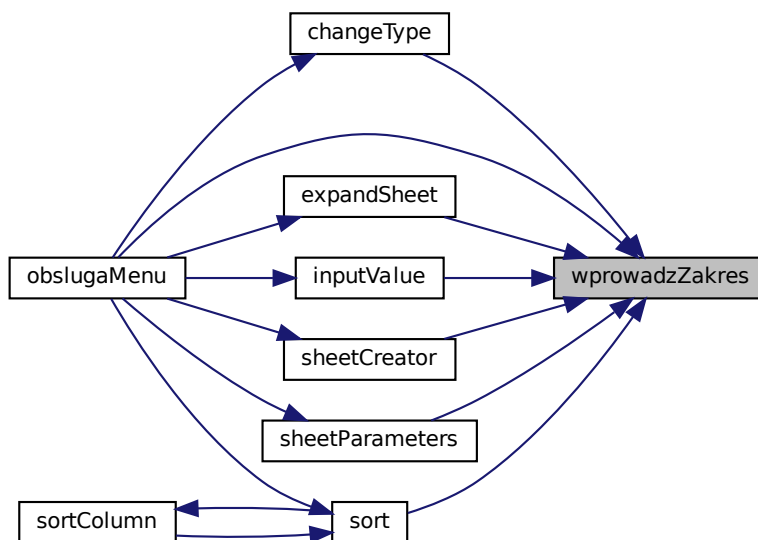
Zwraca

Wartość z zakresu <min; max>

Oto graf wywołań dla tej funkcji:



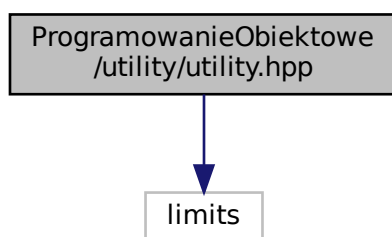
Oto graf wywołań tej funkcji:



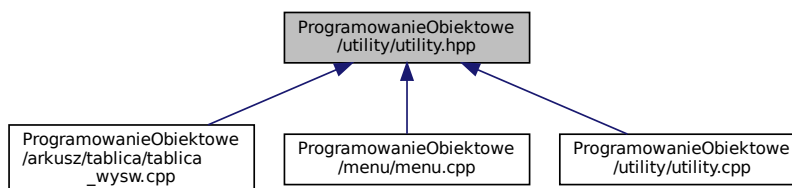
5.25 Dokumentacja pliku ProgramowanieObiektowe/utility/utility.hpp

```
#include <limits>
```

Wykres zależności załączania dla `utility.hpp`:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Funkcje

- `int wprowadzInt ()`
funkcja od wprowadzania wartości typu int
- `int wprowadzZakres (int min=1, int max=std::numeric_limits< int >::max())`
Wprowadź wartość z wyznaczonego zakresu w konsoli Funkcja służy do zwracania wartości wprowadzonej konsolowo, która mieści się wyznaczonym zakresem.
- `void czyscBufor ()`
Funkcja od czyszczenia buforu strumienia CIN.

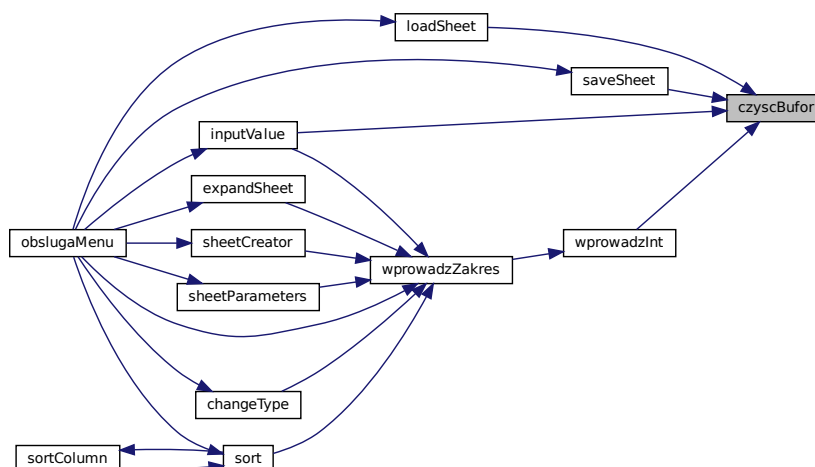
5.25.1 Dokumentacja funkcji

5.25.1.1 czyscBufor()

```
void czyscBufor ( )
```

Funkcja od czyszczenia buforu strumienia CIN.

Funkcja ma za zadanie wyczyścić bufor strumienia wejściowego CIN celem wprowadzenia np. string'a Oto graf wywołań tej funkcji:



5.25.1.2 wprowadzZakres()

```
int wprowadzZakres (
    int min = 1,
    int max = std::numeric_limits< int >::max() )
```

Wprowadź wartość z wyznaczonego zakresu w konsoli Funkcja służy do zwracania wartości wprowadzonej konsolowo, która mieści się wyznaczonym zakresie.

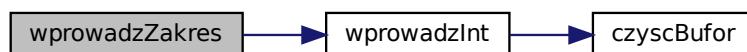
Parametry

in	<i>min</i>	Minimalna wartość jaką można wprowadzić
in	<i>max</i>	Maksymalna wartość jaką można wprowadzić

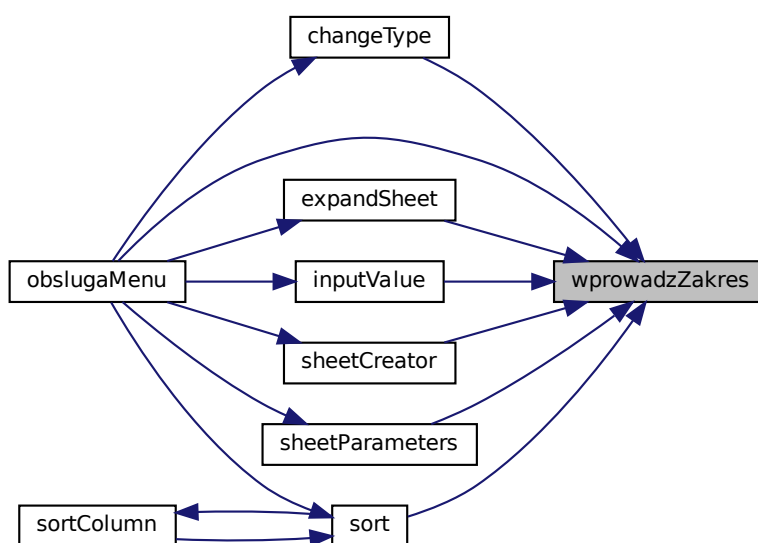
Zwraca

Wartość z zakresu <min; max>

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



Index

BadFileException, 7
begin
 Column, 14

Cell, 8
 operator<, 11
 operator>, 11
 operator+, 10
CellType
 cellType.hpp, 46
cellType.hpp
 CellType, 46
 IntCell, 46
changeType
 menu.cpp, 69
 menu.hpp, 81
Column, 12
 begin, 14
 Column, 13
 end, 14
 generateCellArray, 14
 getCell, 15
 getHeight, 16
 getType, 16
 operator[], 16
 resize, 17
columnParameters
 menu.cpp, 70
 menu.hpp, 82
countCalculateableColumns
 operacje.cpp, 92
 operacje.hpp, 99
createColumnArray
 Sheet, 31
czyscBufor
 utility.cpp, 105
 utility.hpp, 108

DisplaySheet
 tablica_wysw.cpp, 57
 tablica_wysw.hpp, 59

end
 Column, 14
error.hpp
 PLIK_ACCESS, 62
 PLIK_FORMAT, 62
 PLIK_ROZMIAR, 62
 TABLICA_SIZE, 62
 TABLICA_ZAKR, 62

Wyjatki, 61
expandSheet
 menu.cpp, 71
 menu.hpp, 83

fileOperations.cpp
 loadFile, 63
 saveFile, 63
fileOperations.hpp
 loadFile, 65
 saveFile, 66

generateCellArray
 Column, 14
generujMenu
 menu.cpp, 72
 menu.hpp, 83
getCalcValue
 IntCell, 22
 NumericCell, 27
getCell
 Column, 15
getColumn
 Sheet, 32
getHeight
 Column, 16
 Sheet, 33
getType
 Column, 16
getValue
 IntCell, 22
 StringCell, 39
 StringInterface, 43
getWidth
 Sheet, 33

inputValue
 menu.cpp, 72
 menu.hpp, 84
IntCell, 18
 cellType.hpp, 46
 getCalcValue, 22
 getValue, 22
 IntCell, 21
 setValue, 22

loadFile
 fileOperations.cpp, 63
 fileOperations.hpp, 65
loadSheet

- menu.cpp, 73
 - menu.hpp, 85
- maxColumn
 - operacje.cpp, 92
 - operacje.hpp, 100
- maxRow
 - operacje.cpp, 93
 - operacje.hpp, 100
- menu.cpp
 - changeType, 69
 - columnParameters, 70
 - expandSheet, 71
 - generujMenu, 72
 - inputValue, 72
 - loadSheet, 73
 - obslugaMenu, 74
 - rowParameters, 75
 - saveSheet, 76
 - sheetCreator, 77
 - sheetParameters, 78
 - sort, 79
- menu.hpp
 - changeType, 81
 - columnParameters, 82
 - expandSheet, 83
 - generujMenu, 83
 - inputValue, 84
 - loadSheet, 85
 - obslugaMenu, 86
 - rowParameters, 86
 - saveSheet, 87
 - sheetCreator, 88
 - sheetParameters, 89
 - sort, 90
- minColumn
 - operacje.cpp, 94
 - operacje.hpp, 101
- minRow
 - operacje.cpp, 94
 - operacje.hpp, 101
- NotNumericValue, 23
- NumericCell, 24
 - getCalcValue, 27
 - operator<, 28
 - operator>, 28
 - operator+, 27
- obslugaMenu
 - menu.cpp, 74
 - menu.hpp, 86
- operacje.cpp
 - countCalculateableColumns, 92
 - maxColumn, 92
 - maxRow, 93
 - minColumn, 94
 - minRow, 94
 - sortColumn, 95
 - sumColumn, 96
 - sumRow, 96
- operacje.hpp
 - countCalculateableColumns, 99
 - maxColumn, 100
 - maxRow, 100
 - minColumn, 101
 - minRow, 101
 - sortColumn, 102
 - sumColumn, 103
 - sumRow, 104
- operator<
 - Cell, 11
 - NumericCell, 28
 - StringCell, 40
- operator>
 - Cell, 11
 - NumericCell, 28
 - StringCell, 40
- operator+
 - Cell, 10
 - NumericCell, 27
 - StringCell, 39
- operator[]
 - Column, 16
 - Sheet, 34
- PLIK_ACCESS
 - error.hpp, 62
- PLIK_FORMAT
 - error.hpp, 62
- PLIK_ROZMIAR
 - error.hpp, 62
- ProgramowanieObiektowe/arkusz/cell/cell.hpp, 45
- ProgramowanieObiektowe/arkusz/cell/cellType.hpp, 46
- ProgramowanieObiektowe/arkusz/cell/intCell.cpp, 47
- ProgramowanieObiektowe/arkusz/cell/intCell.hpp, 47
- ProgramowanieObiektowe/arkusz/cell/numericCell.cpp, 49
- ProgramowanieObiektowe/arkusz/cell/numericCell.hpp, 49
- ProgramowanieObiektowe/arkusz/cell/stringCell.cpp, 51
- ProgramowanieObiektowe/arkusz/cell/stringCell.hpp, 51
- ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp, 52
- ProgramowanieObiektowe/arkusz/column/column.cpp, 53
- ProgramowanieObiektowe/arkusz/column/column.hpp, 54
- ProgramowanieObiektowe/arkusz/tablica/tablica.cpp, 55
- ProgramowanieObiektowe/arkusz/tablica/tablica.hpp, 56
- ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.cpp, 56
- ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.hpp, 58
- ProgramowanieObiektowe/error.hpp, 60
- ProgramowanieObiektowe/io/fileOperations.cpp, 62
- ProgramowanieObiektowe/io/fileOperations.hpp, 64

ProgramowanieObiektowe/main.cpp, 67
ProgramowanieObiektowe/menu/menu.cpp, 68
ProgramowanieObiektowe/menu/menu.hpp, 80
ProgramowanieObiektowe/operacje/operacje.cpp, 90
ProgramowanieObiektowe/operacje/operacje.hpp, 97
ProgramowanieObiektowe/utility/utility.cpp, 105
ProgramowanieObiektowe/utility/utility.hpp, 107

resize
 Column, 17
 Sheet, 35

rowParameters
 menu.cpp, 75
 menu.hpp, 86

saveFile
 fileOperations.cpp, 63
 fileOperations.hpp, 66

saveSheet
 menu.cpp, 76
 menu.hpp, 87

setValue
 IntCell, 22
 StringCell, 41
 StringInterface, 44

Sheet, 29
 createColumnArray, 31
 getColumn, 32
 getHeight, 33
 getWidth, 33
 operator[], 34
 resize, 35
 Sheet, 30

sheetCreator
 menu.cpp, 77
 menu.hpp, 88

sheetParameters
 menu.cpp, 78
 menu.hpp, 89

sort
 menu.cpp, 79
 menu.hpp, 90

sortColumn
 operacje.cpp, 95
 operacje.hpp, 102

StringCell, 36
 getValue, 39
 operator<, 40
 operator>, 40
 operator+, 39
 setValue, 41
 StringCell, 39

StringInterface, 41
 getValue, 43
 setValue, 44

sumColumn
 operacje.cpp, 96
 operacje.hpp, 103

sumRow
 operacje.cpp, 96
 operacje.hpp, 104

TABLICA_SIZE
 error.hpp, 62

tablica_wysw.cpp
 DisplaySheet, 57

tablica_wysw.hpp
 DisplaySheet, 59

TABLICA_ZAKR
 error.hpp, 62

utility.cpp
 czyscBufor, 105
 wprowadzZakres, 106

utility.hpp
 czyscBufor, 108
 wprowadzZakres, 108

wprowadzZakres
 utility.cpp, 106
 utility.hpp, 108

Wyjatki
 error.hpp, 61