

## Arkusz Kalkulacyjny

Kamil Czop  
Projekt Programowanie Obiektowe



<b>1 Indeks hierarchiczny</b>	<b>1</b>
1.1 Hierarchia klas	1
<b>2 Indeks klas</b>	<b>3</b>
2.1 Lista klas	3
<b>3 Indeks plików</b>	<b>5</b>
3.1 Lista plików	5
<b>4 Dokumentacja klas</b>	<b>7</b>
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 max()	16
4.3.3.8 min()	17
4.3.3.9 operator[]()	17
4.3.3.10 resize()	18
4.3.3.11 sum()	18
4.4 Dokumentacja klasy Display	19
4.4.1 Opis szczegółowy	19
4.4.2 Dokumentacja funkcji składowych	19
4.4.2.1 simpleControl()	20
4.4.2.2 simpleMenu()	20
4.5 Dokumentacja klasy IntCell	21
4.5.1 Opis szczegółowy	23
4.5.2 Dokumentacja konstruktora i destruktor	23
4.5.2.1 IntCell() [1/2]	23

4.5.2.2 IntCell() [2/2]	23
4.5.3 Dokumentacja funkcji składowych	24
4.5.3.1 getCalcValue()	24
4.5.3.2 getValue()	24
4.5.3.3 setValue()	24
4.6 Dokumentacja szablonu klasy List< T >	25
4.6.1 Opis szczegółowy	26
4.6.2 Dokumentacja konstruktora i destruktora	26
4.6.2.1 List()	26
4.6.3 Dokumentacja funkcji składowych	26
4.6.3.1 begin()	26
4.6.3.2 end()	27
4.6.3.3 getElement()	27
4.6.3.4 getSize()	28
4.6.3.5 operator[]()	28
4.6.3.6 pop()	29
4.6.3.7 push()	30
4.7 Dokumentacja struktury NotNumericValue	30
4.7.1 Opis szczegółowy	31
4.8 Dokumentacja klasy NumericCell	32
4.8.1 Opis szczegółowy	33
4.8.2 Dokumentacja funkcji składowych	34
4.8.2.1 getCalcValue()	34
4.8.2.2 operator+()	34
4.8.2.3 operator<()	35
4.8.2.4 operator>()	36
4.9 Dokumentacja klasy Sheet	36
4.9.1 Opis szczegółowy	37
4.9.2 Dokumentacja konstruktora i destruktora	37
4.9.2.1 Sheet()	38
4.9.3 Dokumentacja funkcji składowych	38
4.9.3.1 createColumnArray()	38
4.9.3.2 getColumn()	39
4.9.3.3 getHeight()	40
4.9.3.4 getWidth()	41
4.9.3.5 operator[]()	41
4.9.3.6 resize()	42
4.10 Dokumentacja klasy StringCell	43
4.10.1 Opis szczegółowy	46
4.10.2 Dokumentacja konstruktora i destruktora	46
4.10.2.1 StringCell()	46
4.10.3 Dokumentacja funkcji składowych	46

4.10.3.1	<code>getValue()</code>	46
4.10.3.2	<code>operator+()</code>	47
4.10.3.3	<code>operator&lt;()</code> [1/2]	47
4.10.3.4	<code>operator&lt;()</code> [2/2]	48
4.10.3.5	<code>operator&gt;()</code> [1/2]	48
4.10.3.6	<code>operator&gt;()</code> [2/2]	49
4.10.3.7	<code>setValue()</code>	49
4.11	Dokumentacja klasy <code>StringInterface</code>	50
4.11.1	Opis szczegółowy	52
4.11.2	Dokumentacja funkcji składowych	52
4.11.2.1	<code>getValue()</code>	52
4.11.2.2	<code>setValue()</code>	52
<b>5</b>	<b>Dokumentacja plików</b>	<b>55</b>
5.1	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/cell/cell.hpp</code>	55
5.2	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/cell/cellType.hpp</code>	56
5.2.1	Dokumentacja typów wyliczanych	56
5.2.1.1	<code>CellType</code>	56
5.3	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/cell/intCell.cpp</code>	57
5.4	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/cell/intCell.hpp</code>	57
5.5	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/cell/numericCell.cpp</code>	59
5.6	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/cell/numericCell.hpp</code>	59
5.7	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/cell/stringCell.cpp</code>	61
5.8	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/cell/stringCell.hpp</code>	61
5.9	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp</code>	62
5.10	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/column/column.cpp</code>	63
5.11	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/column/column.hpp</code>	64
5.12	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/tablica/tablica.cpp</code>	65
5.13	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/tablica/tablica.hpp</code>	66
5.14	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.cpp</code>	66
5.14.1	Dokumentacja funkcji	67
5.14.1.1	<code>columnWidth()</code>	67
5.14.1.2	<code>DisplaySheet()</code>	68
5.15	Dokumentacja pliku <code>ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.hpp</code>	69
5.15.1	Dokumentacja funkcji	70
5.15.1.1	<code>columnWidth()</code>	70
5.15.1.2	<code>DisplaySheet()</code>	71
5.16	Dokumentacja pliku <code>ProgramowanieObiektowe/CursesUI/display.hpp</code>	72
5.17	Dokumentacja pliku <code>ProgramowanieObiektowe/error.hpp</code>	73
5.17.1	Dokumentacja typów wyliczanych	74
5.17.1.1	Wyjatk	74
5.18	Dokumentacja pliku <code>ProgramowanieObiektowe/io/fileOperations.cpp</code>	74

5.18.1 Dokumentacja funkcji	75
5.18.1.1 loadFile()	75
5.18.1.2 saveFile()	76
5.19 Dokumentacja pliku ProgramowanieObiektowe/io/fileOperations.hpp	77
5.19.1 Dokumentacja funkcji	78
5.19.1.1 loadFile()	79
5.19.1.2 saveFile()	79
5.20 Dokumentacja pliku ProgramowanieObiektowe/main.cpp	80
5.21 Dokumentacja pliku ProgramowanieObiektowe/menu/menu.cpp	81
5.21.1 Dokumentacja funkcji	82
5.21.1.1 changeType()	82
5.21.1.2 columnParameters()	83
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()	87
5.21.1.9 saveSheet()	88
5.21.1.10 sheetCreator()	89
5.21.1.11 sheetParameters()	90
5.21.1.12 sort()	91
5.22 Dokumentacja pliku ProgramowanieObiektowe/menu/menu.hpp	92
5.22.1 Dokumentacja funkcji	93
5.22.1.1 changeType()	93
5.22.1.2 columnParameters()	94
5.22.1.3 expandSheet()	95
5.22.1.4 generujMenu()	95
5.22.1.5 inputValue()	96
5.22.1.6 loadSheet()	96
5.22.1.7 obslugaMenu()	97
5.22.1.8 rowParameters()	98
5.22.1.9 saveSheet()	99
5.22.1.10 sheetCreator()	100
5.22.1.11 sheetParameters()	101
5.22.1.12 sort()	102
5.23 Dokumentacja pliku ProgramowanieObiektowe/operacje/operacje.cpp	102
5.23.1 Dokumentacja funkcji	103
5.23.1.1 countCalculateableColumns()	104
5.23.1.2 maxRow()	104
5.23.1.3 minRow()	105
5.23.1.4 sortColumn()	106

---

5.23.1.5 sumRow()	107
5.24 Dokumentacja pliku ProgramowanieObiektowe/operacje/operacje.hpp	108
5.24.1 Dokumentacja funkcji	109
5.24.1.1 countCalculateableColumns()	109
5.24.1.2 maxRow()	110
5.24.1.3 minRow()	110
5.24.1.4 sortColumn()	111
5.24.1.5 sumRow()	112
5.25 Dokumentacja pliku ProgramowanieObiektowe/utility/list.hpp	113
5.26 Dokumentacja pliku ProgramowanieObiektowe/utility/utility.cpp	114
5.26.1 Dokumentacja funkcji	114
5.26.1.1 czyscBufor()	114
5.26.1.2 wprowadzZakres()	115
5.27 Dokumentacja pliku ProgramowanieObiektowe/utility/utility.hpp	116
5.27.1 Dokumentacja funkcji	117
5.27.1.1 czyscBufor()	117
5.27.1.2 wprowadzZakres()	118
<b>Index</b>	<b>119</b>





# Chapter 1

## Indeks hierarchiczny

### 1.1 Hierarchia klas

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

Column . . . . .	12
Display . . . . .	19
std::exception	
BadFileException . . . . .	7
NotNumericValue . . . . .	30
List< T > . . . . .	25
Sheet . . . . .	36
StringInterface . . . . .	50
Cell . . . . .	8
NumericCell . . . . .	32
IntCell . . . . .	21
StringCell . . . . .	43



## Chapter 2

# Indeks klas

## 2.1 Lista klas

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

<a href="#">BadFileException</a>	Wyjątek wyrzucany w przypadku braku dostępu. Wyjątek jest wyrzucany gdy użytkownik nie ma dostępu do pliku lub nieistnieje (odczyt) . . . . .	7
<a href="#">Cell</a>	Interfejs/ klasa czysto wirtualna komórka Interfejs komórka posiadający metody ogólne klas dziedziczących . . . . .	8
<a href="#">Column</a>	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
<a href="#">Display</a>	<a href="#">Display</a> Klasa wyświetlacza Curses . . . . .	19
<a href="#">IntCell</a>	<a href="#">IntCell</a> komórka z wartością całkowitą Komórka przyjmująca wartości całkowite . . . . .	21
<a href="#">List&lt; T &gt;</a>	The <a href="#">List</a> Własna implementacja klasy List/Vector . . . . .	25
<a href="#">NotNumericValue</a>	Wyjątek wyrzucany w przypadku wprowadzenia wartości nie numerycznej do komórki. Wyjątek powinien być wyrzucany gdy wprowadzony element nie może być przetworzony na wartość liczbowa . . . . .	30
<a href="#">NumericCell</a>	<a href="#">NumericCell</a> komórki liczbowe Ogólna klasa abstrakcyjna komórek liczbowych które mogą posługiwać się wartościami rzeczywistymi . . . . .	32
<a href="#">Sheet</a>	Klasa opisująca Arkusz Klasa Arkusz przechowywująca tablicę kolumn i jej rozmiar . . . . .	36
<a href="#">StringCell</a>	<a href="#">StringCell</a> Komórka tekstowa Klasa komórki przyjmującej wartości tekstowe . . . . .	43
<a href="#">StringInterface</a>	Intefejs elementów przyjmujących/zwracających elementy string . . . . .	50



## Chapter 3

# Indeks plików

### 3.1 Lista plików

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

ProgramowanieObiektowe/error.hpp	73
ProgramowanieObiektowe/main.cpp	80
ProgramowanieObiektowe/arkusz/cell/cell.hpp	55
ProgramowanieObiektowe/arkusz/cell/cellType.hpp	56
ProgramowanieObiektowe/arkusz/cell/intCell.cpp	57
ProgramowanieObiektowe/arkusz/cell/intCell.hpp	57
ProgramowanieObiektowe/arkusz/cell/numericCell.cpp	59
ProgramowanieObiektowe/arkusz/cell/numericCell.hpp	59
ProgramowanieObiektowe/arkusz/cell/stringCell.cpp	61
ProgramowanieObiektowe/arkusz/cell/stringCell.hpp	61
ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp	62
ProgramowanieObiektowe/arkusz/column/column.cpp	63
ProgramowanieObiektowe/arkusz/column/column.hpp	64
ProgramowanieObiektowe/arkusz/tablica/tablica.cpp	65
ProgramowanieObiektowe/arkusz/tablica/tablica.hpp	66
ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.cpp	66
ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.hpp	69
ProgramowanieObiektowe/CursesUI/controller.hpp	??
ProgramowanieObiektowe/CursesUI/display.hpp	72
ProgramowanieObiektowe/io/fileOperations.cpp	74
ProgramowanieObiektowe/io/fileOperations.hpp	77
ProgramowanieObiektowe/menu/menu.cpp	81
ProgramowanieObiektowe/menu/menu.hpp	92
ProgramowanieObiektowe/operacje/operacje.cpp	102
ProgramowanieObiektowe/operacje/operacje.hpp	108
ProgramowanieObiektowe/utility/list.hpp	113
ProgramowanieObiektowe/utility/utility.cpp	114
ProgramowanieObiektowe/utility/utility.hpp	116



## 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

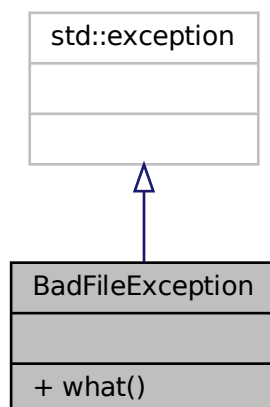
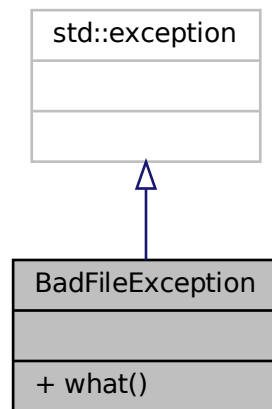


Diagram współpracy dla BadFileException:



## 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

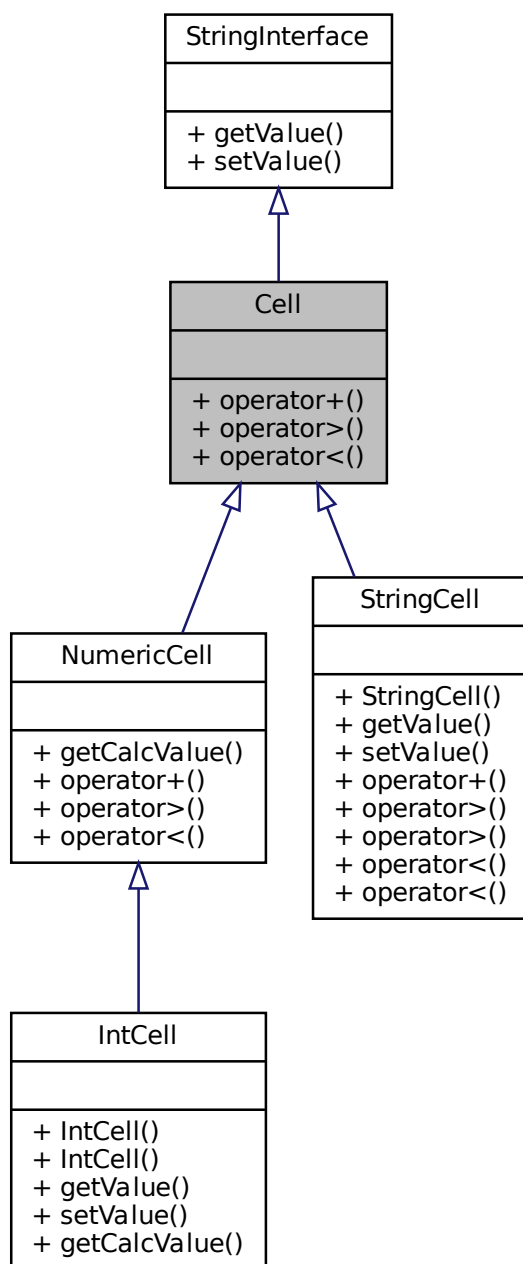
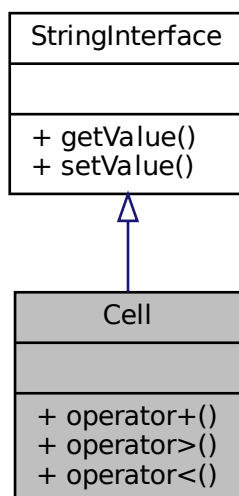


Diagram współpracy dla Cell:



## 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:

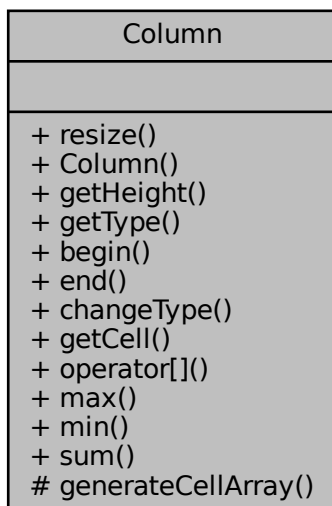
- [ProgramowanieObiektowe/arkusz/cell/cell.hpp](#)

## 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 zwracająca 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
- `double max ()`  
*Funkcja szukania maksymalnej wartości kolumny.*
- `double min ()`  
*Funkcja szukania minimalnej wartości kolumny.*
- `double sum ()`  
*Funkcja licząca sumę elementów 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 destruktora

#### 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

<code>in</code>	<code>height</code>	Rozmiar tablicy komórek w kolumnie
<code>in</code>	<code>type</code>	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

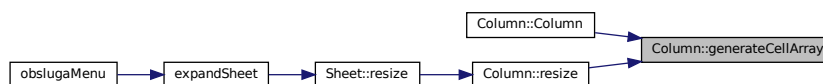
## Wyjątki

<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 max()**

```
double Column::max ( )
```

Funkcja szukania maksymalnej wartości kolumny.

Funkcja przeszukuje całą kolumnę celem określenia największej wprowadzonej wartości

**Zwraca**

Zwraca wartość największą kolumny



#### 4.3.3.8 min()

```
double Column::min ( )
```

Funkcja szukania minimalnej wartości kolumny.

Funkcja przeszukuje całą kolumnę celem określenia najmniejszej wprowadzonej wartości

**Zwraca**

Zwraca wartość najmniejszą kolumny

#### 4.3.3.9 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**

<i>std::out_of_range</i>	Gdy y jest poza zakresem kolumny (y > height)
--------------------------	---

**Zwraca**

referencja komórki z tablicy

Oto graf wywołań dla tej funkcji:



#### 4.3.3.10 `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	<code>newHeight</code>	Nowy rozmiar kolumny
----	------------------------	----------------------

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.3.3.11 `sum()`

```
double Column::sum ( )
```

Funkcja licząca sumę elementów kolumny.

Funkcja zwraca sumę całej kolumny

Zwraca

Zwraca sumę wszystkich elementów kolumny

Dokumentacja dla tej klasy została wygenerowana z plików:

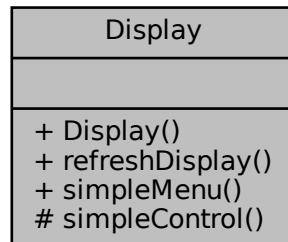
- ProgramowanieObiektowe/arkusz/column/[column.hpp](#)
- ProgramowanieObiektowe/arkusz/column/[column.cpp](#)

## 4.4 Dokumentacja klasy Display

`Display` Klasa wyświetlacza Curses.

```
#include <display.hpp>
```

Diagram współpracy dla Display:



### Metody publiczne

- `Display ()`  
*Display konstruktor domniemany wyświetlacza.*
- void `refreshDisplay ()`  
*refreshDisplay Odświeża w pełni wyświetlacz curses Potrafi wykryć zmianę rozmiaru okna terminala*
- int `simpleMenu ()`  
*simpleMenu Metoda wyświetlająca proste menu*

### Metody chronione

- int `simpleControl ()`  
*simpleControl Metoda obsługi klawiatury w prostym menu*

#### 4.4.1 Opis szczegółowy

`Display` Klasa wyświetlacza Curses.

#### 4.4.2 Dokumentacja funkcji składowych

#### 4.4.2.1 simpleControl()

```
int Display::simpleControl ( ) [protected]
```

simpleControl Metoda obsługi klawiatury w prostym menu

Zwraca

wartości 1 i -1 służą do poruszania strzałkami w górę i w dół, wartość 0 w przypadku kliknięcia klawisza enter

Oto graf wywoływań tej funkcji:



#### 4.4.2.2 simpleMenu()

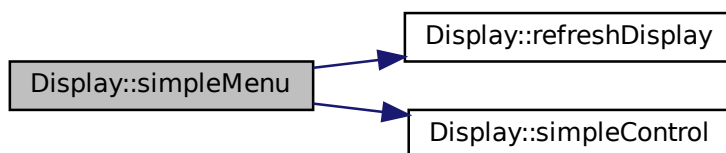
```
int Display::simpleMenu ( )
```

simpleMenu Metoda wyświetlająca proste menu

Zwraca

Indeks wybranego elementu

Oto graf wywołań dla tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- ProgramowanieObiektowe/CursesUI/[display.hpp](#)
- ProgramowanieObiektowe/CursesUI/[display.cpp](#)

## 4.5 Dokumentacja klasy IntCell

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

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

Diagram dziedziczenia dla IntCell

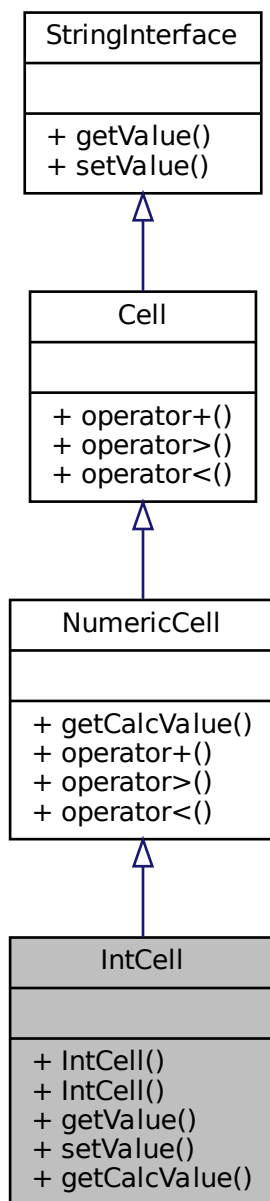
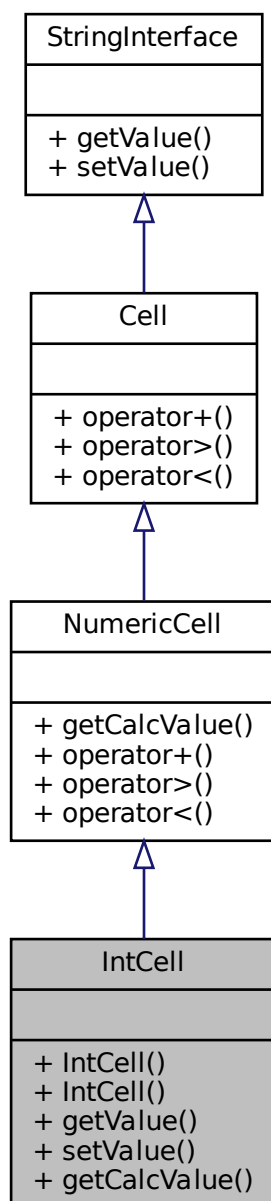


Diagram współpracy dla IntCell:



## 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

### 4.5.1 Opis szczegółowy

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

### 4.5.2 Dokumentacja konstruktora i destruktora

#### 4.5.2.1 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
----	-------	----------------------------

#### 4.5.2.2 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
----	-------	----------------------------

### 4.5.3 Dokumentacja funkcji składowych

#### 4.5.3.1 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](#).

#### 4.5.3.2 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](#).

#### 4.5.3.3 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

<a href="#">NotNumericValue</a>	Brak możliwości przetworzenia wartości tekstowej na liczbową
---------------------------------	--

##### Parametry

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



Implementuje [StringInterface](#).

Dokumentacja dla tej klasy została wygenerowana z plików:

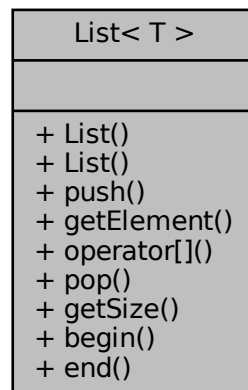
- ProgramowanieObiektowe/arkusz/cell/[intCell.hpp](#)
- ProgramowanieObiektowe/arkusz/cell/[intCell.cpp](#)

## 4.6 Dokumentacja szablonu klasy List< T >

The [List](#) Własna implementacja klasy List/Vector.

```
#include <list.hpp>
```

Diagram współpracy dla List< T >:



### Metody publiczne

- [List](#) ()  
*List konstruktor domniemany listy.*
- [List](#) (std::initializer\_list< T > elements)  
*List Konstruktor listy z początkowymi elementami.*
- void [push](#) (T element)  
*push Dodawanie elementów do listy Metoda służąca dodawaniu elementów do istniejącej listy elementów typu T*
- T [getElement](#) (size\_t index)  
*getElement Getter elementu z listy Zwraca element zapisany w tablicy listy pod danym adresem.*
- T [operator\[\]](#) (size\_t index)  
*operator [] Wrapper na getter elementów z tablicy Zwraca element zapisany w tablicy listy pod danym adresem.*
- void [pop](#) ()  
*pop Metoda "kasująca" ostatni element z listy Zmniejsza licznik elementów w liście*
- int [getSize](#) ()  
*getSize Getter na ilość elementów w liście*
- T \* [begin](#) ()  
*begin Początek tablicy*
- T \* [end](#) ()  
*end Koniec tablicy*

### 4.6.1 Opis szczegółowy

```
template<typename T>
class List< T >
```

The [List](#) Własna implementacja klasy List/Vector.

### 4.6.2 Dokumentacja konstruktora i destruktora

#### 4.6.2.1 List()

```
template<typename T >
List< T >::List (
    std::initializer_list< T > elements ) [inline]
```

[List](#) Konstruktor listy z początkowymi elementami.

#### Parametry

in	<i>elements</i>	lista inicjalizacyjna elementów typu listy
----	-----------------	--

Oto graf wywołań dla tej funkcji:



### 4.6.3 Dokumentacja funkcji składowych

#### 4.6.3.1 begin()

```
template<typename T >
T* List< T >::begin ( ) [inline]
```

begin Początek tablicy

#### Zwraca

Wskaźnik na początek tablicy elementów

#### 4.6.3.2 end()

```
template<typename T >
T* List< T >::end ( ) [inline]
```

end Koniec tablicy

##### Zwraca

Wskaźnik na koniec tablicy elementów

#### 4.6.3.3 getElement()

```
template<typename T >
T List< T >::getElement (
    size_t index ) [inline]
```

getElement Getter elementu z listy Zwraca element zapisany w tablicy listy pod danym adresem.

##### Parametry

in	<i>index</i>	Indeks elementu w tablicy listy
----	--------------	---------------------------------

##### Wyjątki

<i>std::out_of_range</i>	Wyrzucany gdy param index jest poza zakresem ilości elementów
--------------------------	---

##### Zwraca

Element znajdujący się pod danym indeksem w tablicy

Oto graf wywoływań tej funkcji:



#### 4.6.3.4 getSize()

```
template<typename T >
int List< T >::getSize ( ) [inline]
```

getSize Getter na ilość elementów w liście

##### Zwraca

Zwraca ilość elementów obecnie znajdujących się w liście

#### 4.6.3.5 operator[]()

```
template<typename T >
T List< T >::operator[] (
    size_t index ) [inline]
```

operator [] Wrapper na getter elementów z tablicy Zwraca element zapisany w tablicy listy pod danym adresem.

##### Parametry

in	<i>index</i>	Indeks elementu w tablicy listy
----	--------------	---------------------------------

##### Wyjątki

<i>std::out_of_range</i>	Wyrzucany gdy param index jest poza zakresem ilości elementów
--------------------------	---

##### Zwraca

Element znajdujący się pod danym indeksem w tablicy

Oto graf wywołań dla tej funkcji:



#### 4.6.3.6 pop()

```
template<typename T >  
void List< T >::pop ( ) [inline]
```

pop Metoda "kasująca" ostatni element z listy Zmniejsza licznik elementów w liście

## Wyjątki

<code>std::bad_array_new_length</code>	w przypadku próby redukcji licznika poniżej 0
--	---

## 4.6.3.7 push()

```
template<typename T >
void List< T >::push (
    T element ) [inline]
```

push Dodawanie elementów do listy Metoda służąca dodawaniu elementów do istniejącej listy elementów typu T

## Parametry

in	<i>element</i>	Dodawany element do listy
----	----------------	---------------------------

Oto graf wywoływań tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z pliku:

- ProgramowanieObiektowe/utility/[list.hpp](#)

## 4.7 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

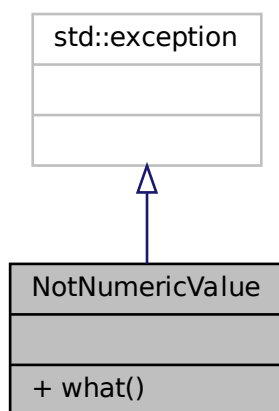
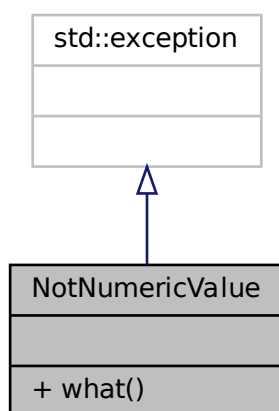


Diagram współpracy dla NotNumericValue:



## Metody publiczne

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

### 4.7.1 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](#)

## 4.8 Dokumentacja klasy NumericCell

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

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

Diagram dziedziczenia dla NumericCell

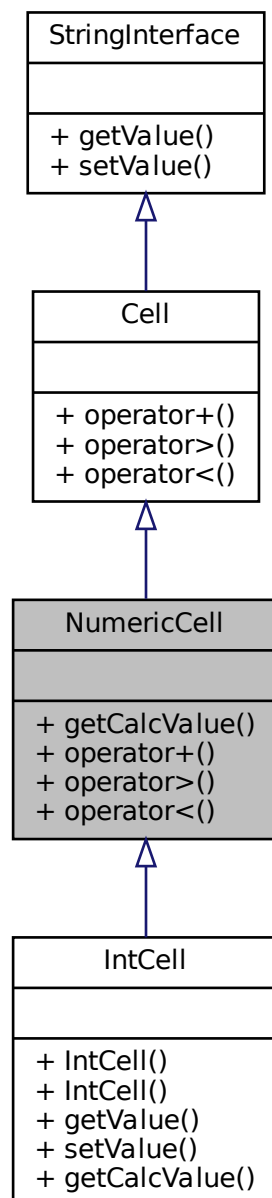
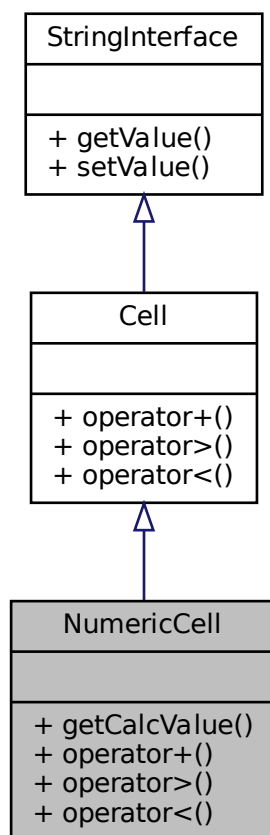




Diagram współpracy dla NumericCell:



## 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

### 4.8.1 Opis szczegółowy

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

## 4.8.2 Dokumentacja funkcji składowych

### 4.8.2.1 getCalcValue()

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

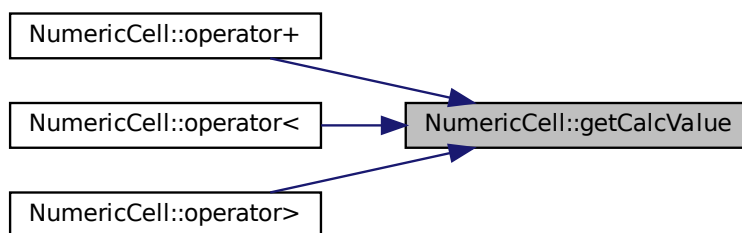
getCalcValue Metoda klasy umożliwiająca bezpośrednie uzyskanie wartości double Metoda umożliwia uzyskanie wartości bezpośrednie bez potrzeby parsowania wartości

#### Zwraca

wartość komórki

Implementowany w [IntCell](#).

Oto graf wywoływań tej funkcji:



### 4.8.2.2 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:

**4.8.2.3 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:



#### 4.8.2.4 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:

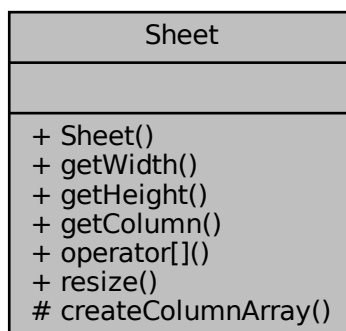
- [ProgramowanieObiektowe/arkusz/cell/numericCell.hpp](#)
- [ProgramowanieObiektowe/arkusz/cell/numericCell.cpp](#)

## 4.9 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.*

### 4.9.1 Opis szczegółowy

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

### 4.9.2 Dokumentacja konstruktora i destruktor

#### 4.9.2.1 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.

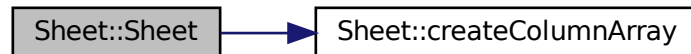
##### Parametry

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:



### 4.9.3 Dokumentacja funkcji składowych

#### 4.9.3.1 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

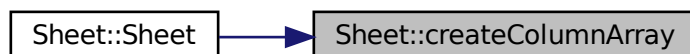
## Wyjątki

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

## Zwraca

Tworzy nową tablicę wskaźników kolumn o wyznaczonych rozmiarach

Oto graf wywoływań tej funkcji:



## 4.9.3.2 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:



#### 4.9.3.3 getHeight()

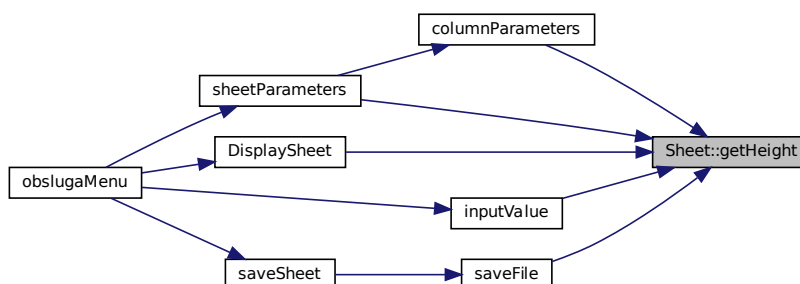
```
size_t Sheet::getHeight ( )
```

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

Zwraca

wysokość arkusza / ilość wierszy

Oto graf wywoływań tej funkcji:





## 4.9.3.4 getWidth()

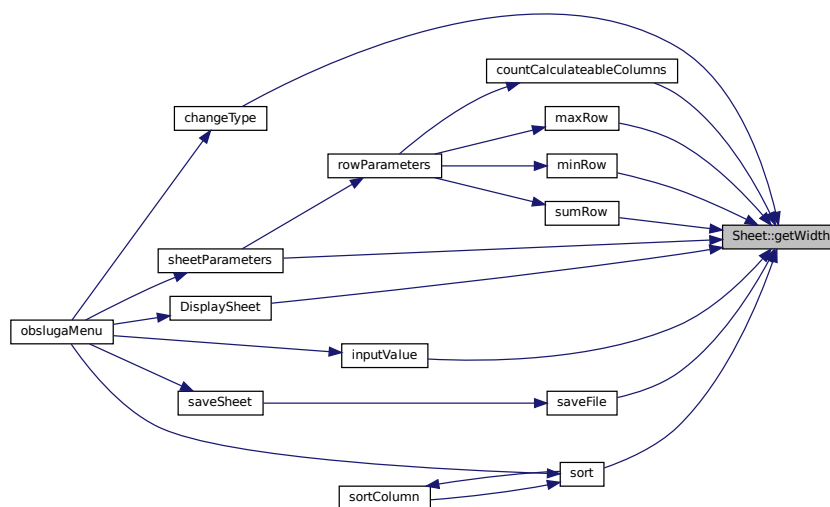
```
size_t Sheet::getWidth ( )
```

getWidth getter szerokości Zwraca ilość kolumn w arkuszu

**Zwraca**

ilość kolumn

Oto graf wywoływań tej funkcji:



## 4.9.3.5 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**

std::out_of_range	Gdy x jest poza zakresem arkusza (x > width)
-------------------	--

**Zwraca**

Referencja na wybraną kolumnę

Oto graf wywołań dla tej funkcji:

**4.9.3.6 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.

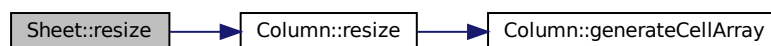
**Parametry**

in	x	Nowa szerokość arkusza
in	y	Nowa wysokość kolumn arkusza

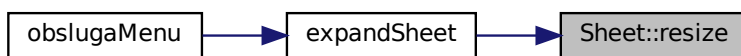
**Wyjątki**

<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](#)

## 4.10 Dokumentacja klasy StringCell

**StringCell** Komórka tekstowa Klasa komórki przyjmującej wartości tekstowe.

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

Diagram dziedziczenia dla StringCell

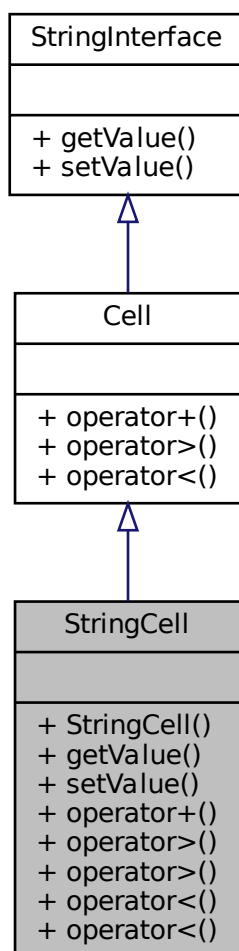
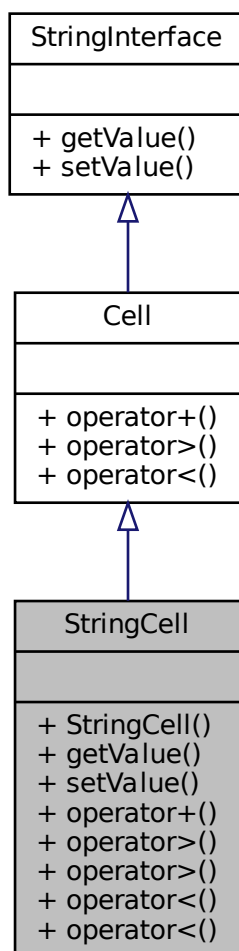


Diagram współpracy dla StringCell:



## Metody publiczne

- **StringCell** (std::string value="?")  
*StringCell* Konstruktor domyślny z opcjonalnym parametrem.
- 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.
- bool **operator>** (StringCell &)  
*operator >* Operator porównania dla komórek tekstowych
- bool **operator>** (Cell &)  
*operator >* operator porównania pozostałych typów komórek.

- bool `operator<` (`StringCell` &)  
*operator < Operator porównania dla komórek tekstowych*
- bool `operator<` (`Cell` &)  
*operator < operator porównania pozostałych typów komórek*

### 4.10.1 Opis szczegółowy

`StringCell` Komórka tekstowa Klasa komórki przyjmującej wartości tekstowe.

### 4.10.2 Dokumentacja konstruktora i destruktora

#### 4.10.2.1 `StringCell()`

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

`StringCell` Konstruktor domyślny z opcjonalnym parametrem.

Konstruktor umożliwiający określenie początkowej wartości komórki.

Parametry

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

### 4.10.3 Dokumentacja funkcji składowych

#### 4.10.3.1 `getValue()`

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

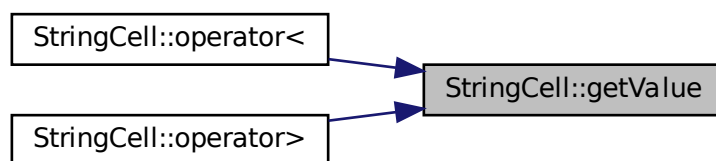
`getValue` Metoda ogólna od uzyskiwania wartości komórki Metoda zwraca wartość w postaci wartości tekstowej komórki

Zwraca

zawartość tekstowa komórki

Implementuje `StringInterface`.

Oto graf wywoływań tej funkcji:



#### 4.10.3.2 `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](#).

#### 4.10.3.3 `operator<()` [1/2]

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

`operator <` operator porównania pozostałych typów komórek

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

Wartość logiczna True

Implementuje [Cell](#).

**4.10.3.4 operator<()** [2/2]

```
bool StringCell::operator< (  
    StringCell & rhs )
```

operator < Operator porównania dla komórek tekstowych

Operator porównania czy obecny obiekt jest większy od drugiej komórki.

## Zwraca

Czy obecna komórka jest niżej leksykalnie.

Oto graf wywołań dla tej funkcji:

**4.10.3.5 operator>()** [1/2]

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

operator > operator porównania pozostałych typów komórek.

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

Wartość logiczna True

Implementuje [Cell](#).

**4.10.3.6 operator>() [2/2]**

```
bool StringCell::operator> (  
    StringCell & rhs )
```

operator > Operator porównania dla komórek tekstowych

Operator porównania czy obecny obiekt jest większy od drugiej komórki.

## Zwraca

Czy obecna komórka jest wyżej leksykalnie

Oto graf wywołań dla tej funkcji:

**4.10.3.7 setValue()**

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

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

## Parametry

in	<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](#)

## 4.11 Dokumentacja klasy StringInterface

Intefejs elementów przyjmujących/zwracających elementy string.

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

Diagram dziedziczenia dla StringInterface

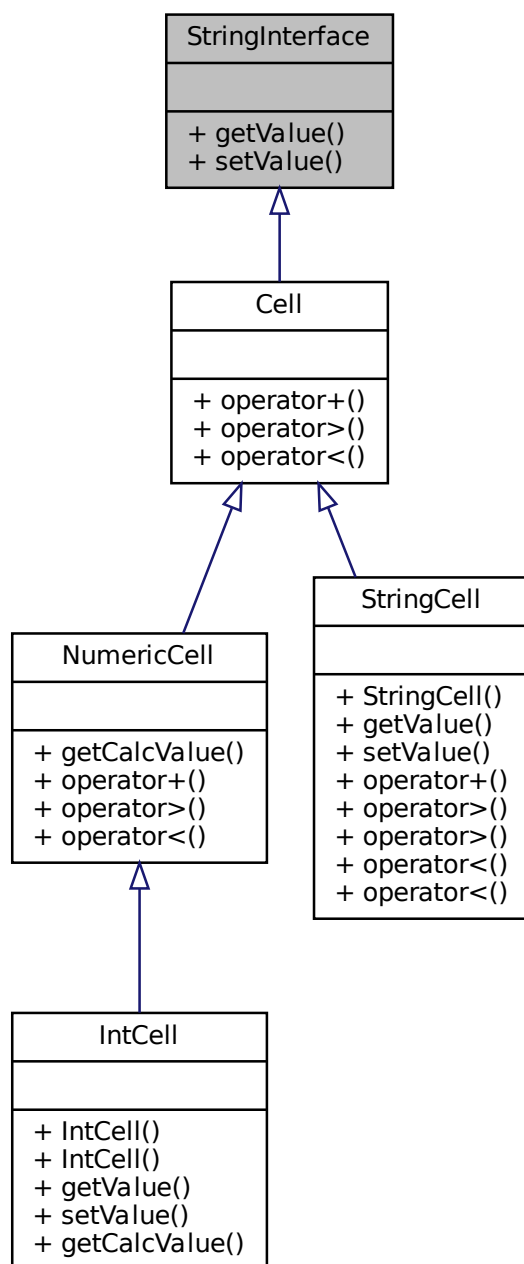
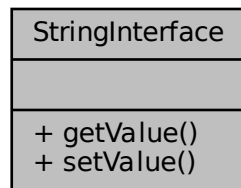


Diagram współpracy dla StringInterface:



## 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*

### 4.11.1 Opis szczegółowy

Intefejs elementów przyjmujących/zwracających elementy string.

### 4.11.2 Dokumentacja funkcji składowych

#### 4.11.2.1 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](#).

#### 4.11.2.2 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](#)



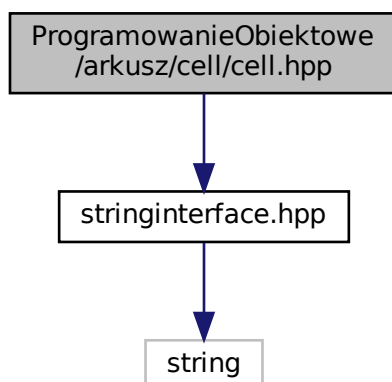
## Chapter 5

# Dokumentacja plików

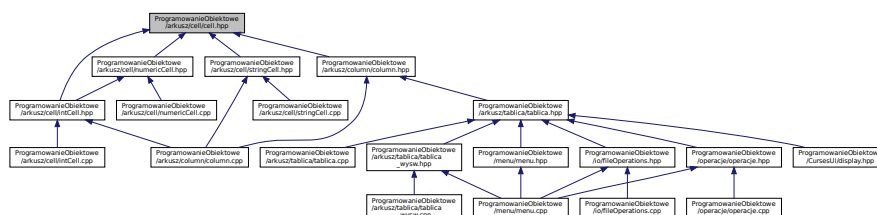
### 5.1 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/cell.hpp

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

Wykres zależności załączania dla cell.hpp:



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



## Komponenty

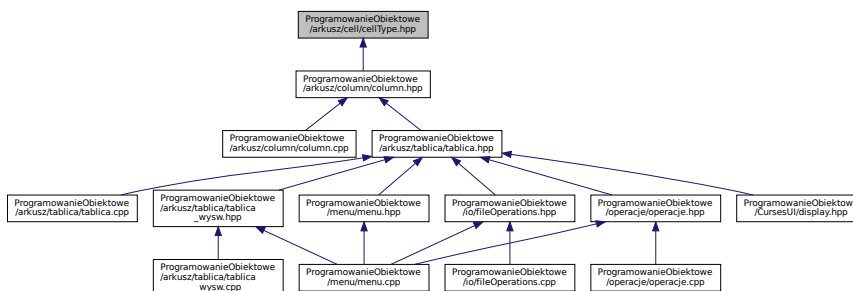
- class `Cell`

*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.
---------	-------------------

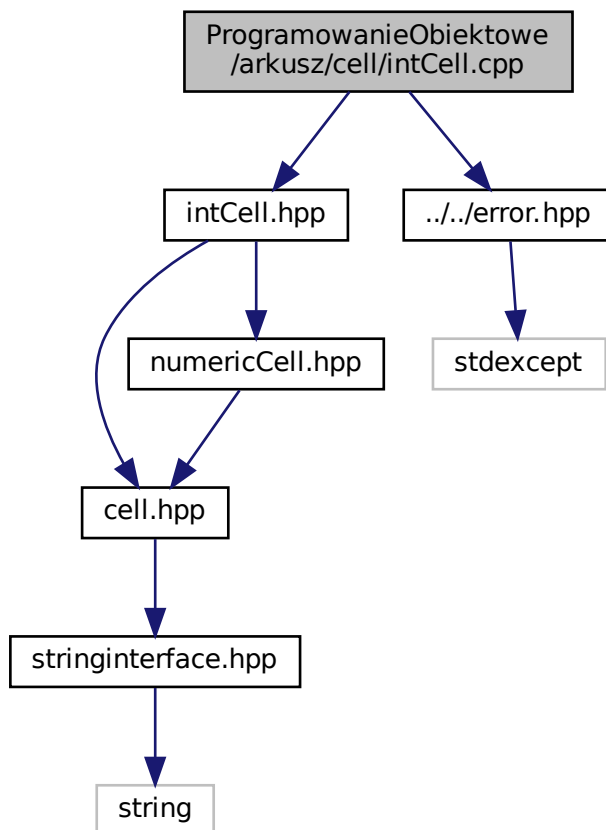


### 5.3 Dokumentacja pliku

#### ProgramowanieObiektowe/arkusz/cell/intCell.cpp

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

Wykres zależności załączania dla intCell.cpp:

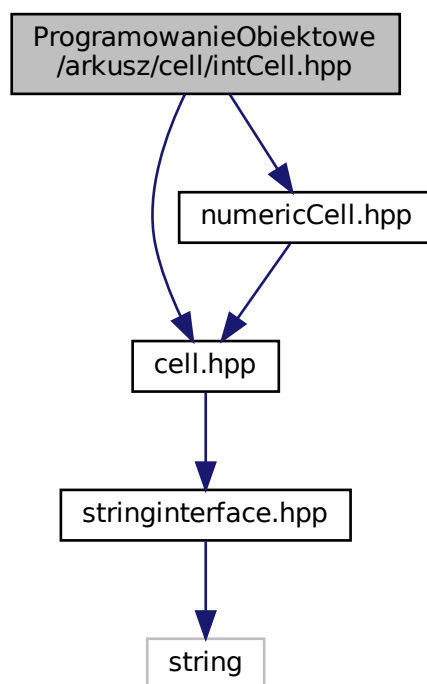


### 5.4 Dokumentacja pliku

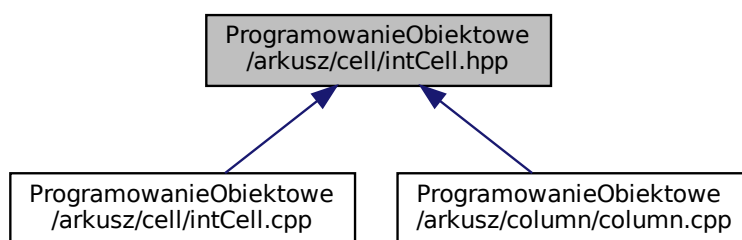
#### ProgramowanieObiektowe/arkusz/cell/intCell.hpp

```
#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](#)

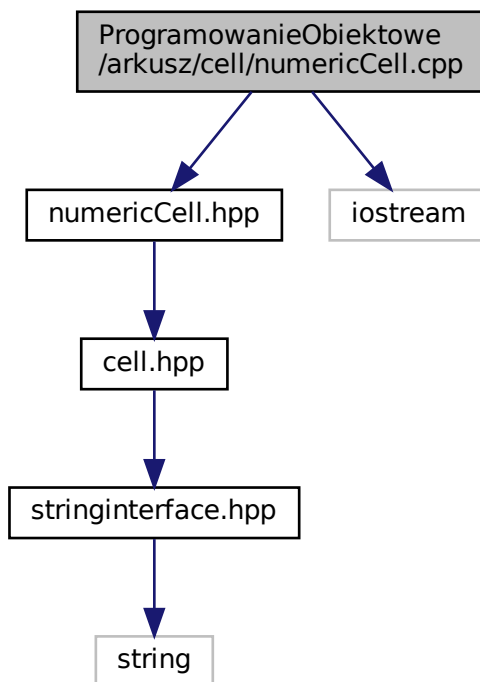
*[IntCell](#) komórka z wartością całkowitą Komórka przyjmująca wartości całkowite.*

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

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

```
#include <iostream>
```

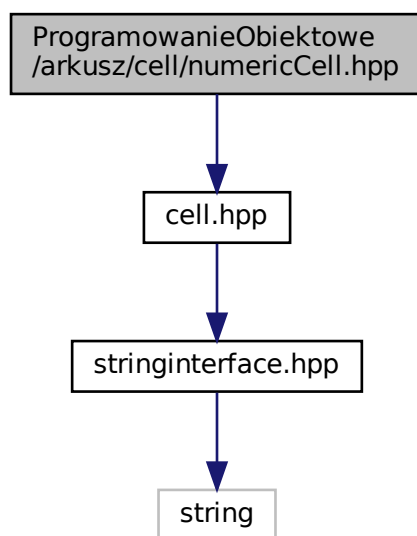
Wykres zależności załączania dla numericCell.cpp:



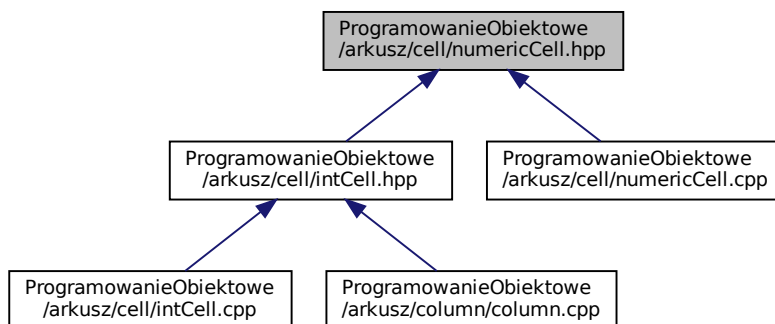
## 5.6 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/numericCell.hpp

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

Wykres zależności załączania dla numericCell.hpp:



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



## Komponenty

- class [NumericCell](#)

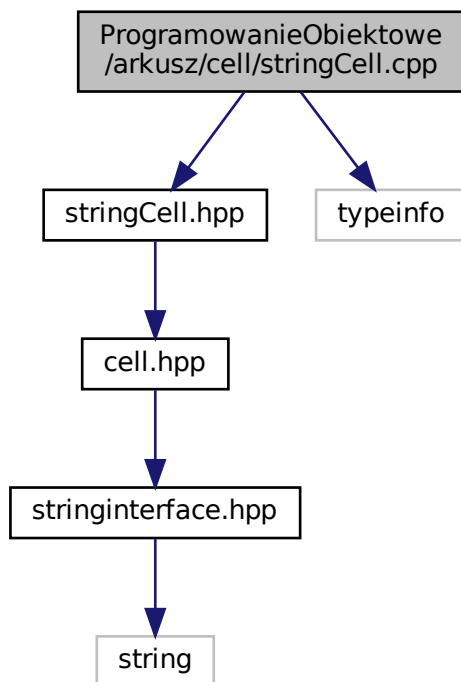
*[NumericCell](#) komórki liczbowe Ogólna klasa abstrakcyjna komórek liczbowych które mogą posługiwać się wartościami rzeczywistymi.*

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

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

```
#include <typeinfo>
```

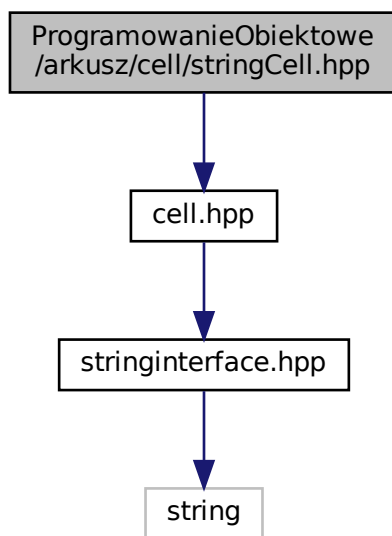
Wykres zależności załączania dla stringCell.cpp:



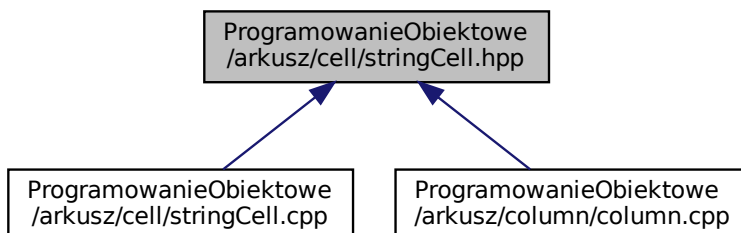
## 5.8 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/stringCell.hpp

```
#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.

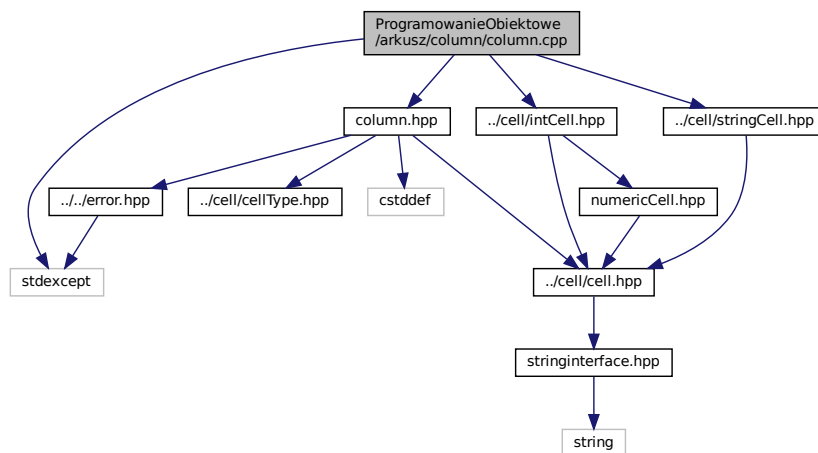
## 5.9 Dokumentacja pliku

### ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp

```
#include <string>
```



Wykres zależności załączania dla column.cpp:



## 5.11 Dokumentacja pliku

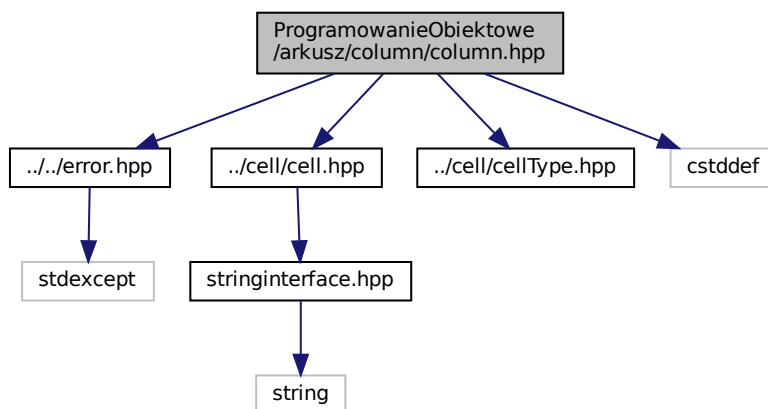
### ProgramowanieObiektowe/arkusz/column/column.hpp

```

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

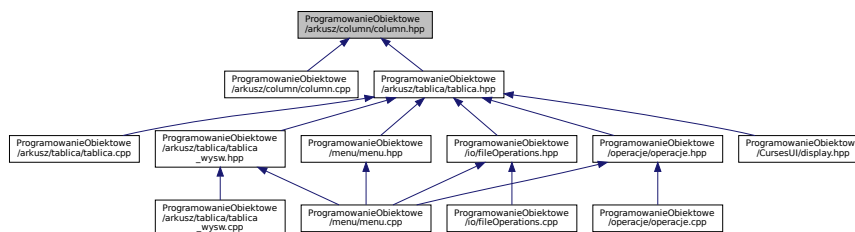
```

Wykres zależności załączania dla column.hpp:





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



## Komponenty

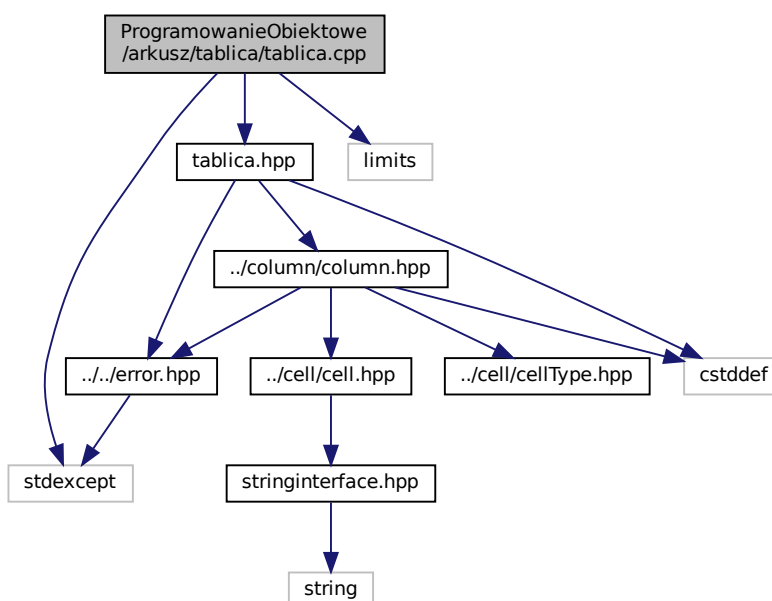
- class [Column](#)

*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:

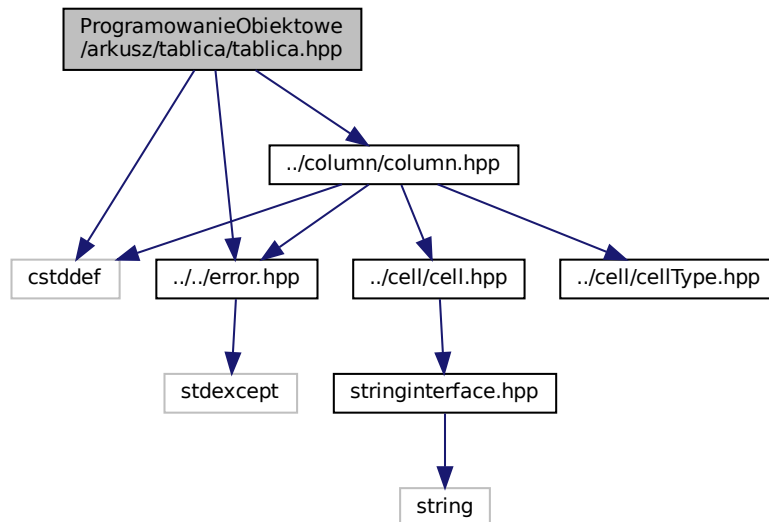


## 5.13 Dokumentacja pliku

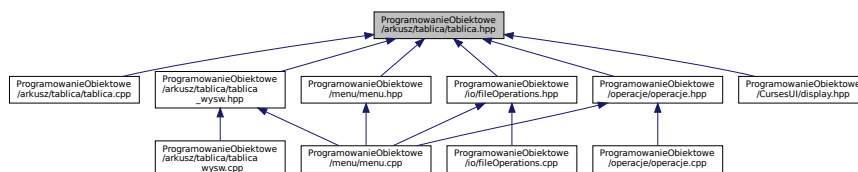
### ProgramowanieObiektowe/arkusz/tablica/tablica.hpp

```
#include <cstdint>
#include "../error/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 przechowująca tablicę kolumn i jej rozmiar.*

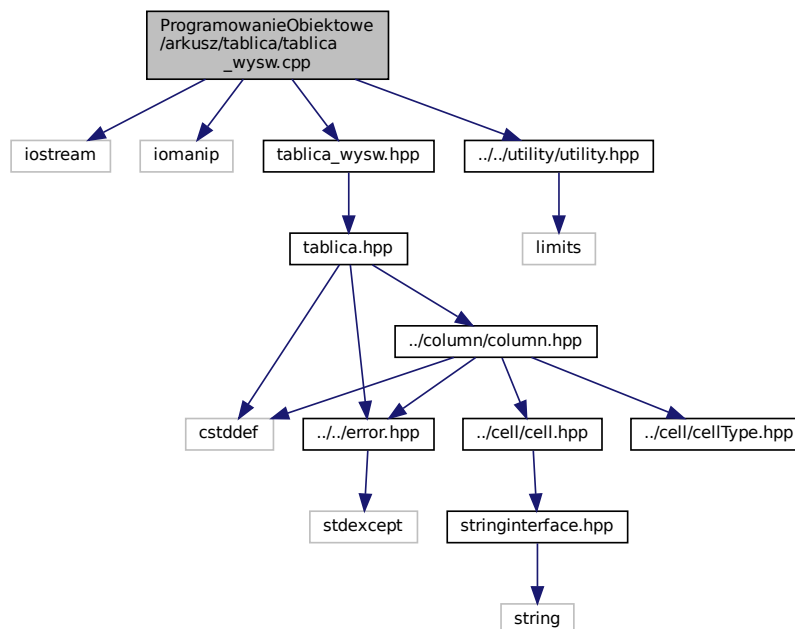
## 5.14 Dokumentacja pliku

### ProgramowanieObiektowe/arkusz/tablica/tablica\_wysw.cpp

```
#include <iostream>
#include <iomanip>
```

```
#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.*
- size\_t **columnWidth** (**Column** column)  
*columnWidth Metoda od uzyskiwania szerokości tekstu w kolumnie*

### 5.14.1 Dokumentacja funkcji

#### 5.14.1.1 columnWidth()

```
size_t columnWidth (
    Column column )
```

columnWidth Metoda od uzyskiwania szerokości tekstu w kolumnie

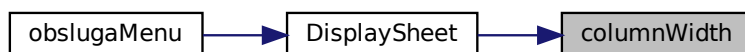
#### Parametry

in	column	Kolumna której długość tekstów będzie sprawdzana
----	--------	--

Zwraca

długość najszerszego tekstu

Oto graf wywołań tej funkcji:



#### 5.14.1.2 DisplaySheet()

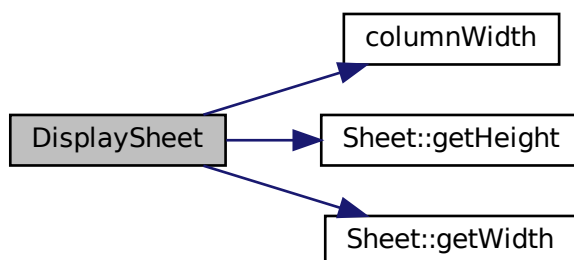
```
void DisplaySheet (  
    Sheet sheet )
```

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:

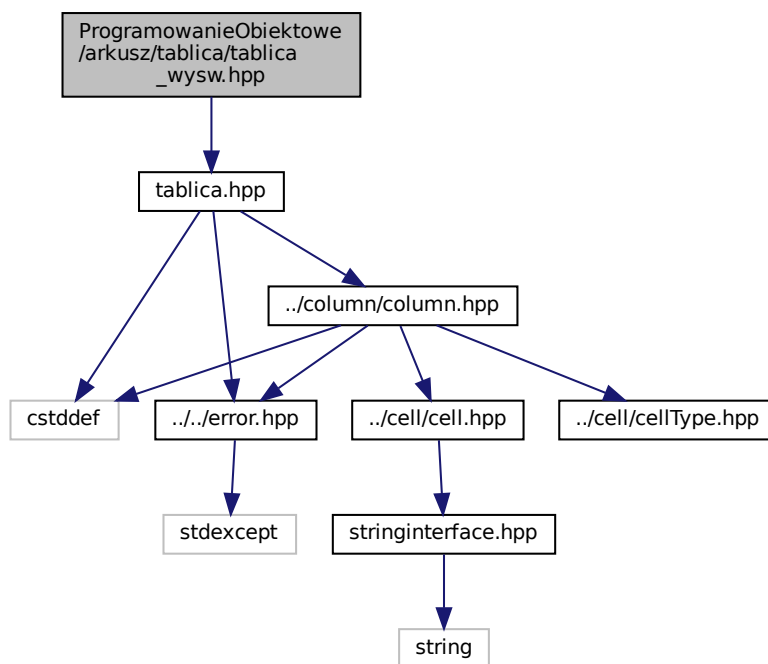


## 5.15 Dokumentacja pliku

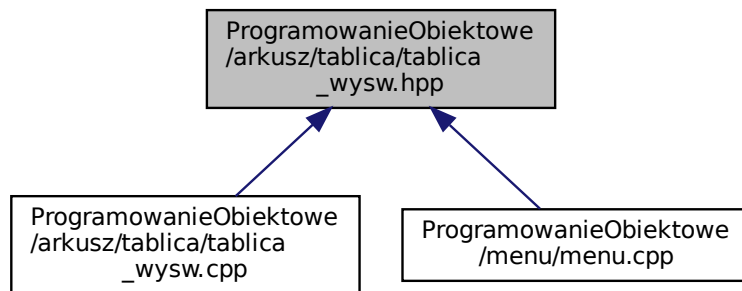
### ProgramowanieObiektowe/arkusz/tablica/tablica\_wysw.hpp

```
#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.*
- size\_t [columnWidth](#) ([Column](#) column)  
*columnWidth Metoda od uzyskiwania szerokości tekstu w kolumnie*

### 5.15.1 Dokumentacja funkcji

#### 5.15.1.1 columnWidth()

```
size_t columnWidth (
    Column column )
```

columnWidth Metoda od uzyskiwania szerokości tekstu w kolumnie

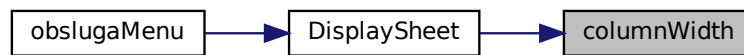
#### Parametry

in	<i>column</i>	Kolumna której długość tekstów będzie sprawdzana
----	---------------	--

Zwraca

długość najszerszego tekstu

Oto graf wywoływań tej funkcji:



### 5.15.1.2 DisplaySheet()

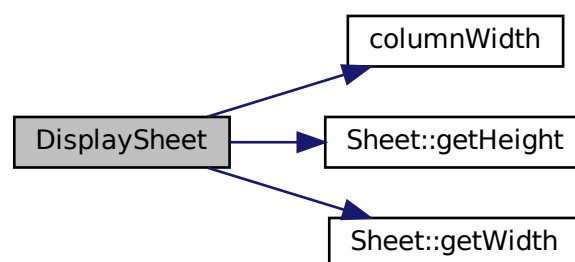
```
void DisplaySheet (  
    Sheet sheet )
```

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 wywołań tej funkcji:

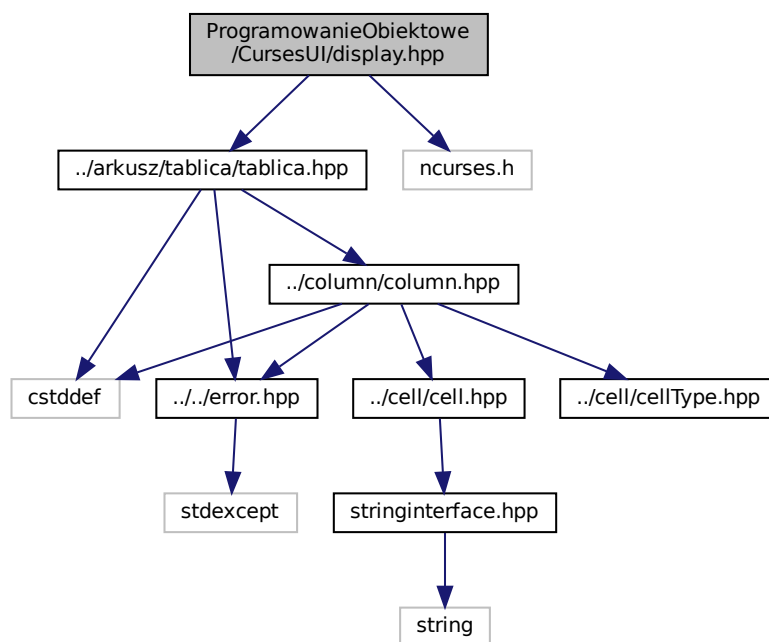


## 5.16 Dokumentacja pliku

### ProgramowanieObiektowe/CursesUI/display.hpp

```
#include "../arkusz/tablica/tablica.hpp"
#include <ncurses.h>
```

Wykres zależności załączania dla display.hpp:



## Komponenty

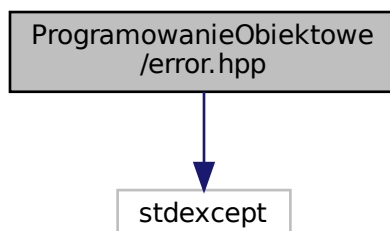
- class [Display](#)  
[Display](#) Klasa wyświetlacza Curses.



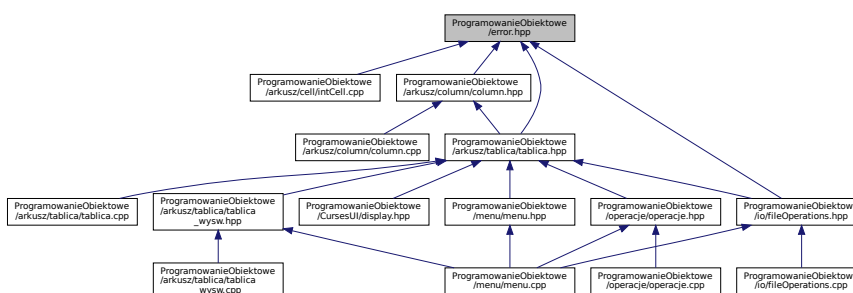
## 5.17 Dokumentacja pliku ProgramowanieObiektowe/error.hpp

```
#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 [Wyjatki](#) : 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.*

### 5.17.1 Dokumentacja typów wyliczanych

#### 5.17.1.1 Wyjątki

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

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

Wartości wyliczeń

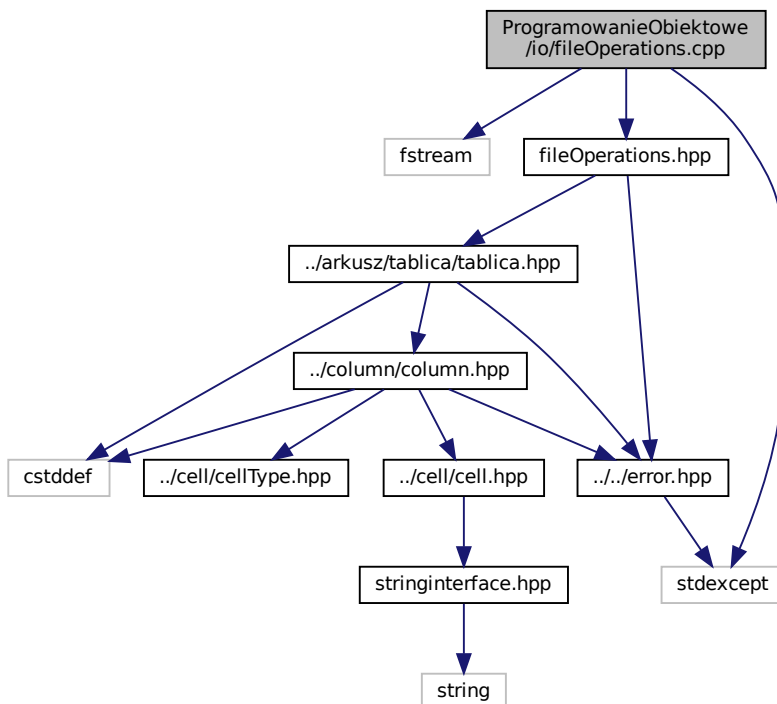
TABLICA_SIZE	Brak błędów.
TABLICA_ZAKR	Próba dostępu do elementu poza zakresem tablicy.
PLIK_ACCESS	Próba utworzenia tablicy o niepoprawnym rozmiarze.
PLIK_FORMAT	Niepoprawna nazwa lub brak dostępu do pliku.
PLIK_ROZMIAR	Niepoprawny format wczytywanego pliku.

## 5.18 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.*

### 5.18.1 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.

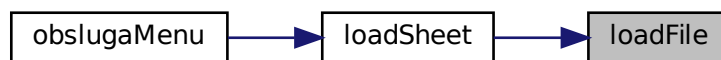
## Parametry

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

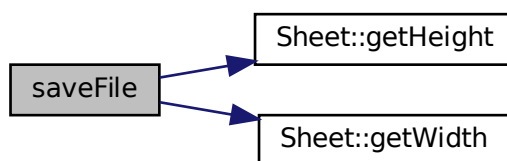
## Parametry

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

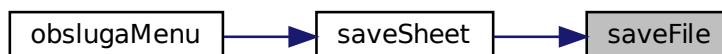
## Wyjątki

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

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

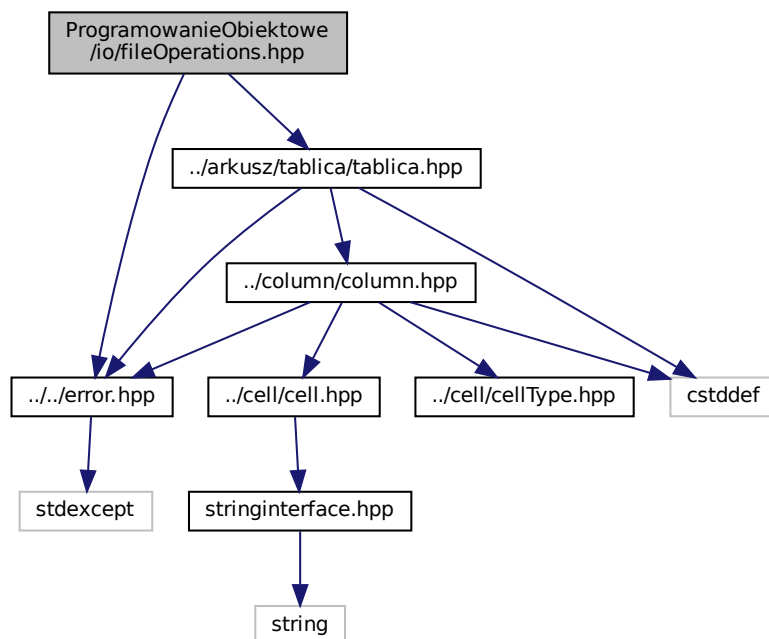


## 5.19 Dokumentacja pliku

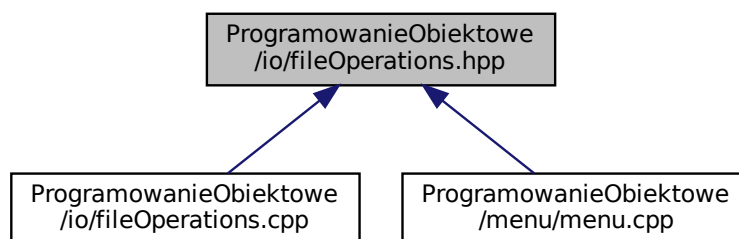
### ProgramowanieObiektowe/io/fileOperations.hpp

```
#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.*

### 5.19.1 Dokumentacja funkcji

### 5.19.1.1 loadFile()

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

Funkcja wczytywania tablicy z pliku.

Funkcja wykonuje wczytanie arkusza z wybranego pliku.

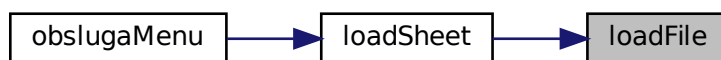
#### Parametry

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:



### 5.19.1.2 saveFile()

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

Funkcja zapisu do pliku.

Funkcja wykonuje zapis do wybranego przez nas pliku

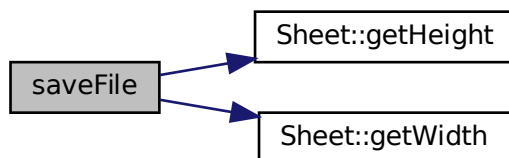
#### Parametry

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

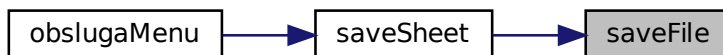
## Wyjątki

<a href="#"><i>BadFileException</i></a>	W przypadku braku dostępu do zapisu
---	-------------------------------------

Oto graf wywołań dla tej funkcji:



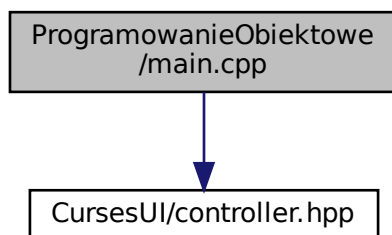
Oto graf wywoływań tej funkcji:



## 5.20 Dokumentacja pliku ProgramowanieObiektowe/main.cpp

```
#include "CursesUI/controller.hpp"
```

Wykres zależności załączania dla main.cpp:





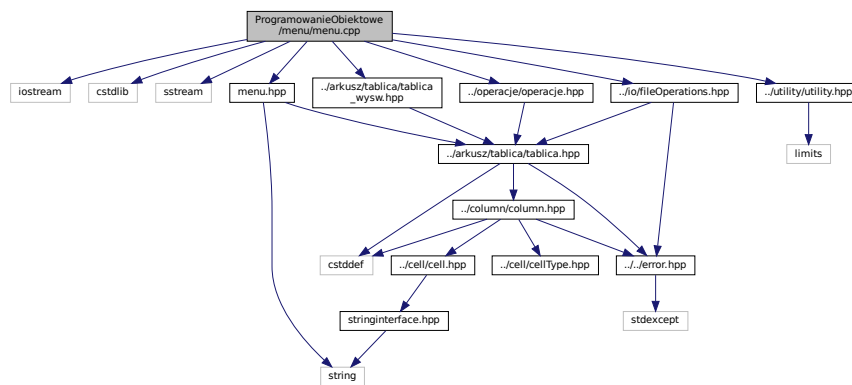
## Funkcje

- **int main ()**

## 5.21 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.*

## 5.21.1 Dokumentacja funkcji

### 5.21.1.1 `changeType()`

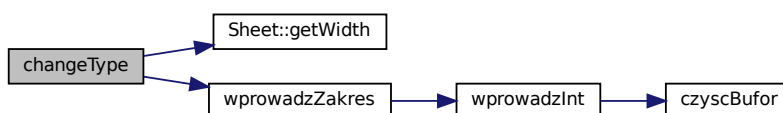
```
void changeType (
    Sheet * arkusz )
```

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 wywoływań tej funkcji:



### 5.21.1.2 columnParameters()

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

Funkcja od wyznaczania parametrów kolumny arkusza.

Funkcja zwraca w postaci tekstowej wszystkie parametry wybranej kolumny

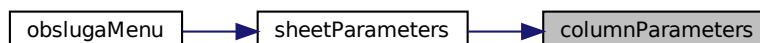
#### Parametry

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:



### 5.21.1.3 expandSheet()

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

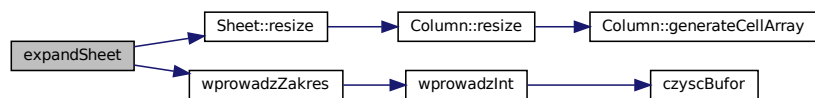
Funkcja modyfikująca rozmiar arkusza.

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 wywoływań tej funkcji:



#### 5.21.1.5 inputValue()

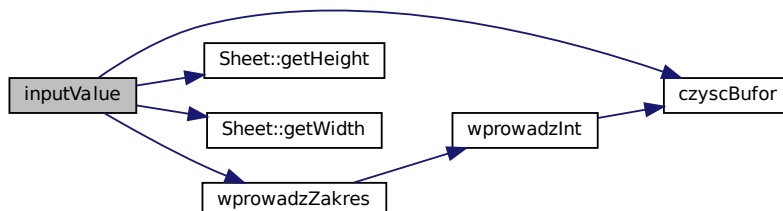
```
void inputValue (
    Sheet * arkusz )
```

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

## Parametry

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

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



## 5.21.1.6 loadSheet()

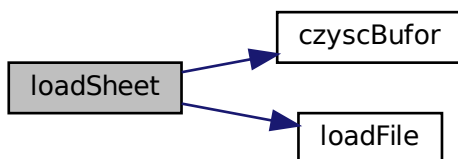
```
void loadSheet (
    Sheet * arkusz )
```

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

## Parametry

<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:



#### 5.21.1.7 obsługaMenu()

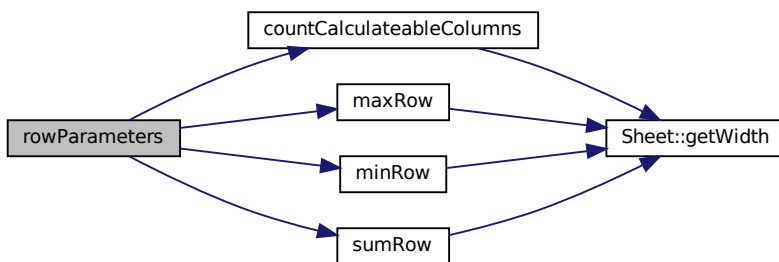
```
void obsługaMenu ( )
```

Funkcja kontrolująca działanie programu.

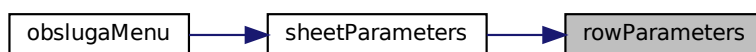
Funkcja zajmująca się obsługą menu programu zarządza tym co będzie wywoływane Oto graf wywołań dla tej



Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 5.21.1.9 saveSheet()

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

Funkcja menu od zapisu.

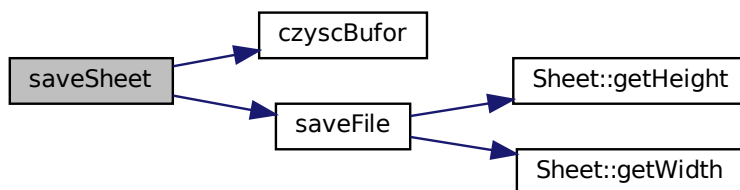
Funkcja menu od zapisu która ma za zadanie przetworzenie i opakowanie funkcji IO zapisPliku

##### 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 )
```

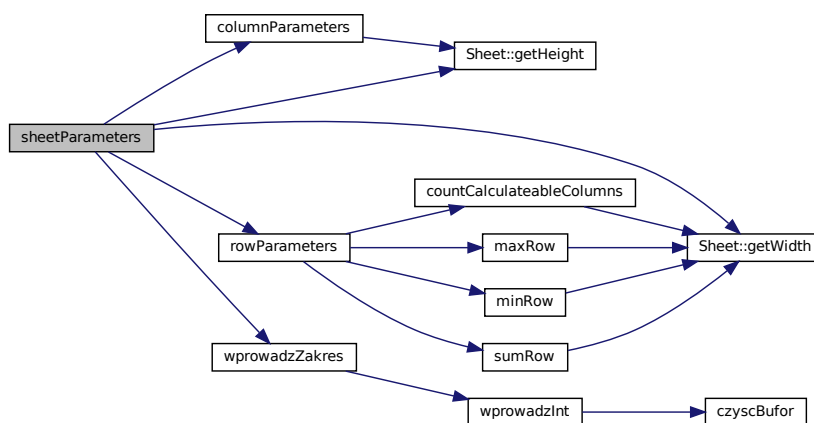
Funkcja menu od wyboru względem czego wyznacza parametry.

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()

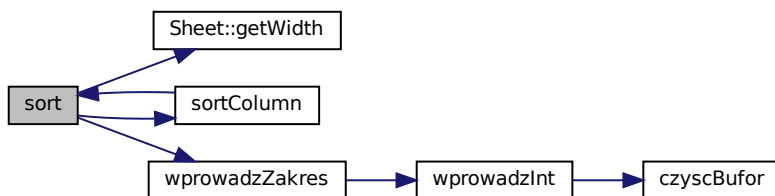
```
void sort (
    Sheet * arkusz )
```

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

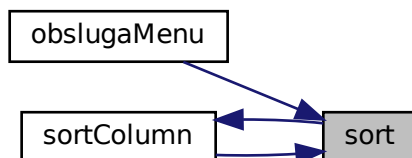
##### Parametry

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

Oto graf wywołań dla tej funkcji:



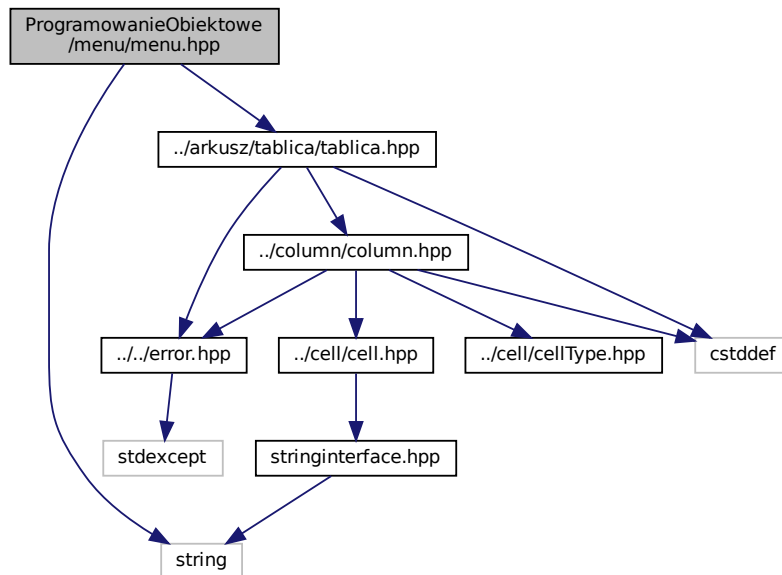
Oto graf wywoływań tej funkcji:



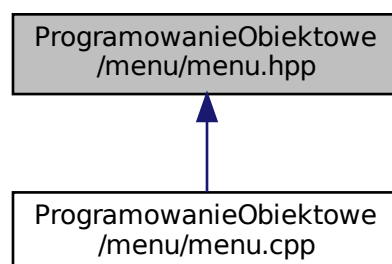
## 5.22 Dokumentacja pliku ProgramowanieObiektowe/menu/menu.hpp

```
#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.*

## 5.22.1 Dokumentacja funkcji

### 5.22.1.1 changeType()

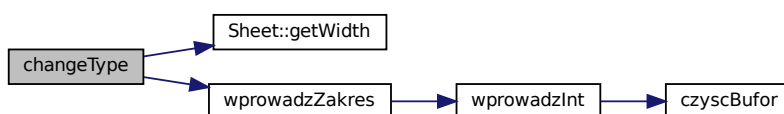
```
void changeType (
    Sheet * arkusz )
```

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:



### 5.22.1.2 columnParameters()

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

Funkcja od wyznaczania parametrów kolumny arkusza.

Funkcja zwraca w postaci tekstowej wszystkie parametry wybranej kolumny

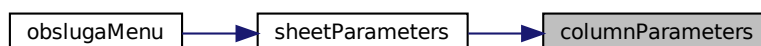
#### Parametry

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:



### 5.22.1.3 expandSheet()

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

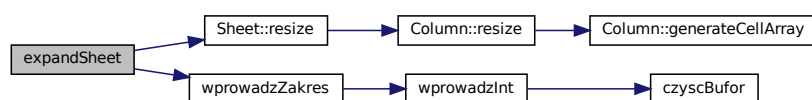
Funkcja modyfikująca rozmiar arkusza.

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:



### 5.22.1.4 generujMenu()

```
void generujMenu ( )
```

Funkcja tworząca menu.

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



### 5.22.1.5 inputValue()

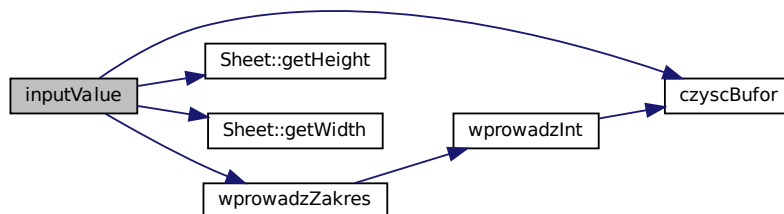
```
void inputValue (
    Sheet * arkusz )
```

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:



### 5.22.1.6 loadSheet()

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

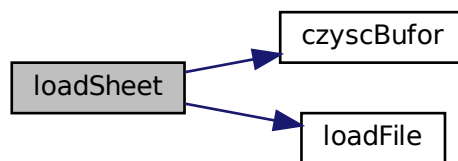
Funkcja wczytywania arkusza Funkcja menu od wczytywania arkusza, ma za zadanie opakowanie funkcji IO wczytajPlik.

#### 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:



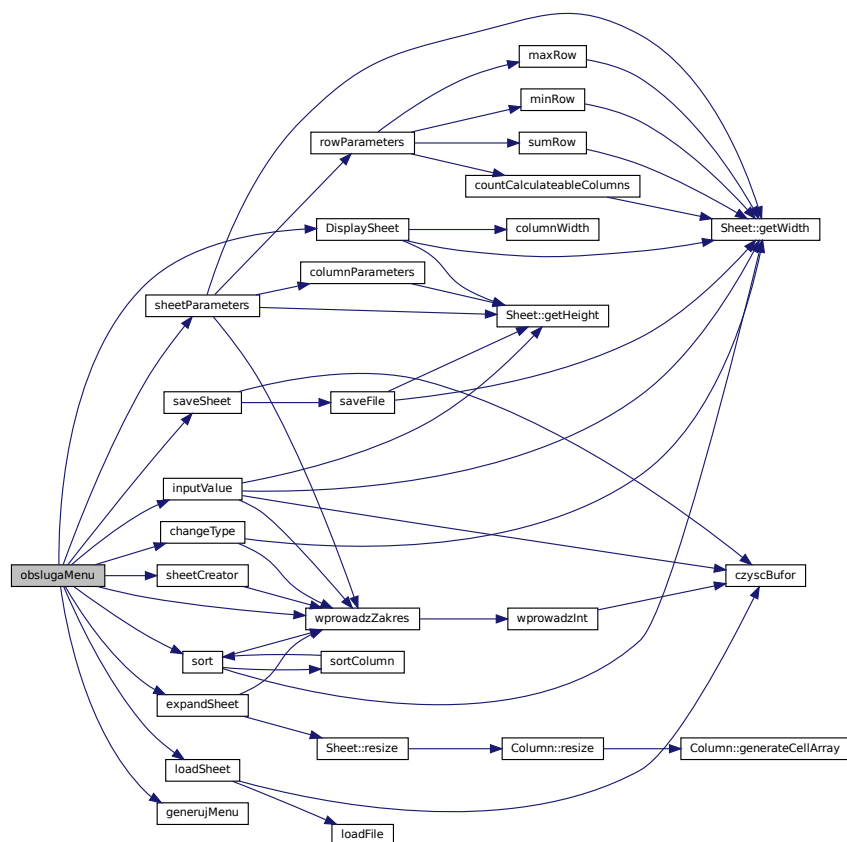
#### 5.22.1.7 obsługaMenu()

```
void obsługaMenu ( )
```

Funkcja kontrolująca działanie programu.

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

funkcji:



### 5.22.1.8 rowParameters()

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

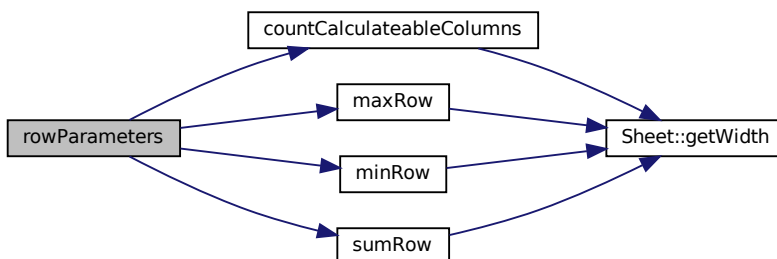
Funkcja od wyznaczania parametrów wiersza arkusza.

Funkcja zwraca w postaci tekstowej wszystkie parametry wybranego wiersza

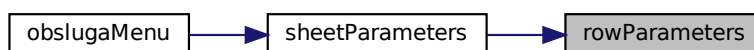
#### Parametry

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:



#### 5.22.1.9 saveSheet()

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

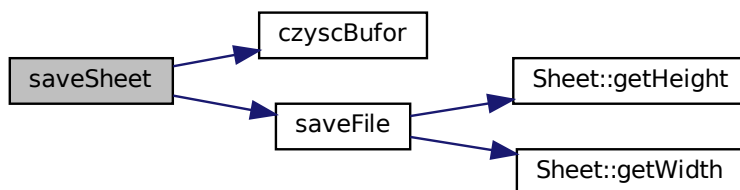
Funkcja menu od zapisu.

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:



#### 5.22.1.10 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:



#### 5.22.1.11 sheetParameters()

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

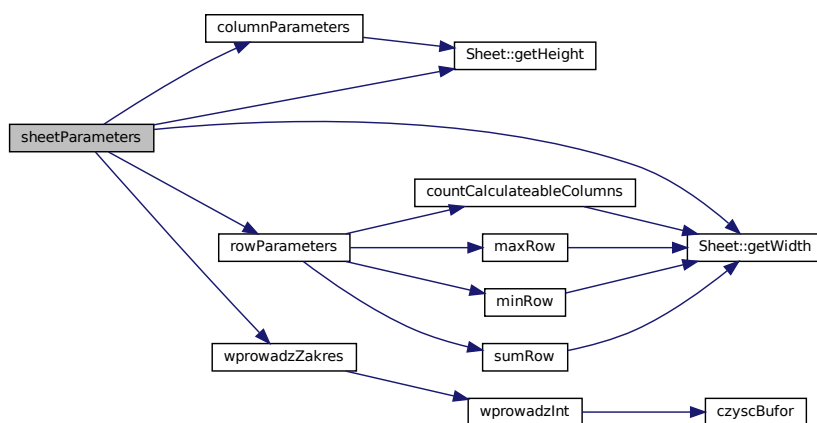
Funkcja menu od wyboru względem czego wyznacza parametry.

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:



#### 5.22.1.12 sort()

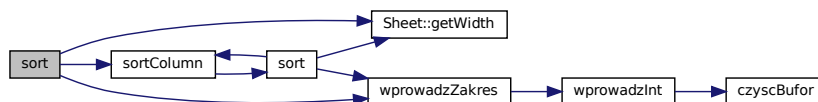
```
void sort (
    Sheet * arkusz )
```

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

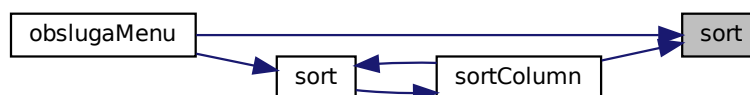
##### Parametry

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

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



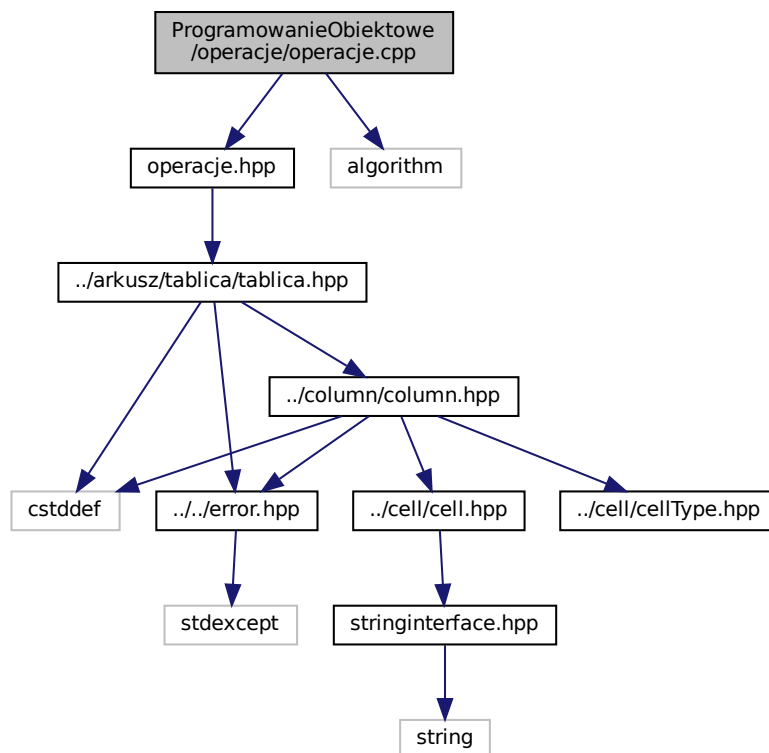
## 5.23 Dokumentacja pliku

### 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.*
- 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*

### 5.23.1 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
--------------	--------------------------------------

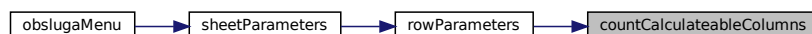
#### Zwraca

liczba kolumn typów obliczalnych

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



### 5.23.1.2 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



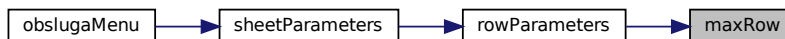
**Zwraca**

Zwraca wartość maksymalną wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

**5.23.1.3 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

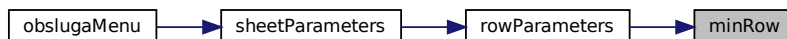
**Zwraca**

Zwraca wartość najmniejszą wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

**5.23.1.4 sortColumn()**

```

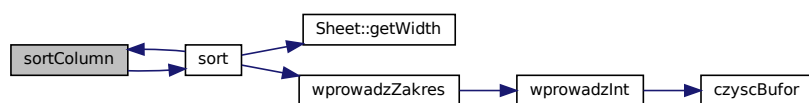
void sortColumn (
    Column * column,
    bool descending = false )
  
```

**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:



### 5.23.1.5 sumRow()

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

Funkcja licząca sumę elementów wiersza.

Funkcja zwraca sumę całego wiersza

#### Parametry

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

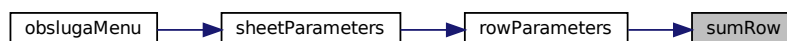
#### Zwraca

Zwraca sumę wszystkich elementów wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

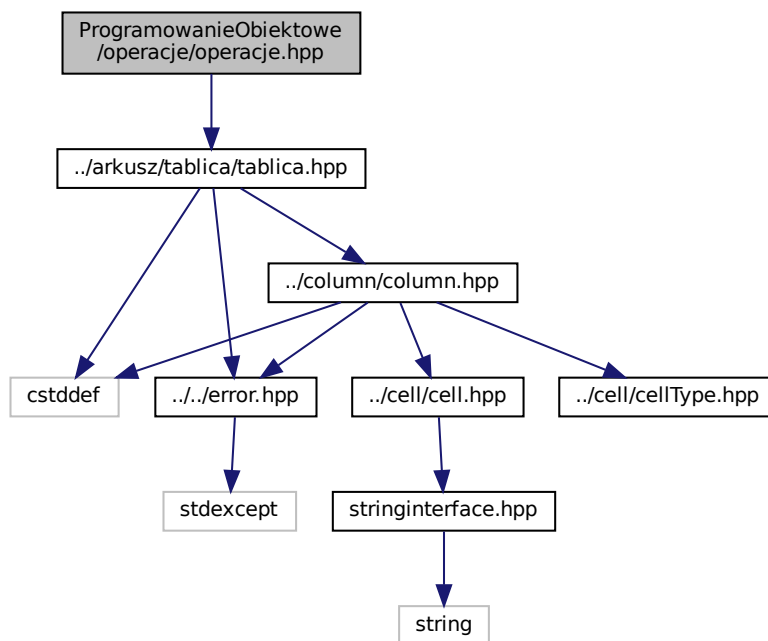


## 5.24 Dokumentacja pliku

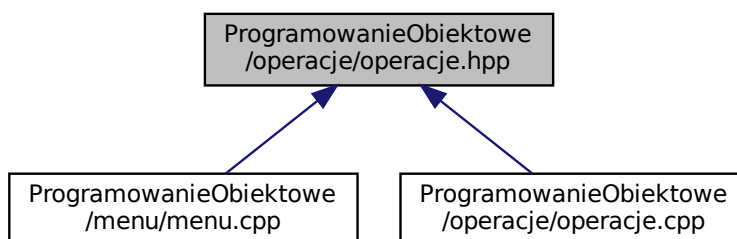
### ProgramowanieObiektowe/operacje/operacje.hpp

```
#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*
- void `sortColumn` (`Column` \*column, bool descending=false)  
*sortKolumna Sortuje komórki w kolumnie rosnąco lub malejąco Sortuje kolumnę w zależności od podanego parametru descending*

## 5.24.1 Dokumentacja funkcji

### 5.24.1.1 countCalculateableColumns()

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

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

#### Parametry

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

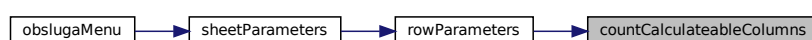
#### Zwraca

liczba kolumn typów obliczalnych

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



### 5.24.1.2 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

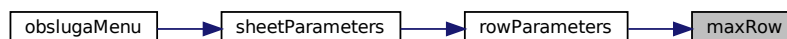
#### Zwraca

Zwraca wartość maksymalną wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



### 5.24.1.3 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

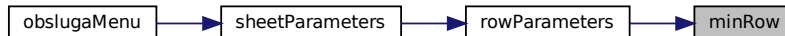
**Zwraca**

Zwraca wartość najmniejszą wiersza

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

**5.24.1.4 sortColumn()**

```

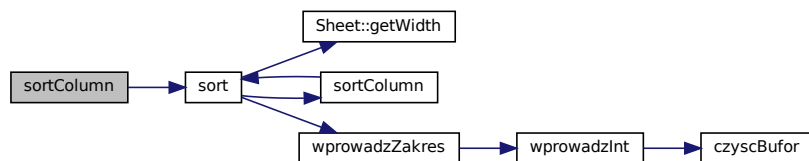
void sortColumn (
    Column * column,
    bool descending = false )
  
```

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

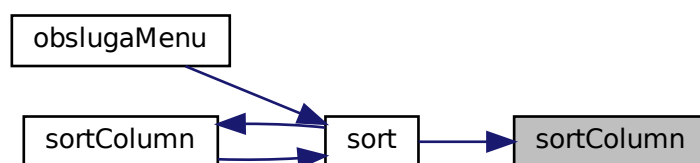
**Parametry**

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:



#### 5.24.1.5 sumRow()

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

Funkcja licząca sumę elementów wiersza.

Funkcja zwraca sumę całego wiersza

##### Parametry

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



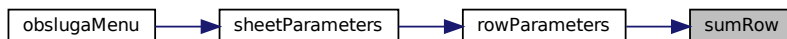
**Zwraca**

Zwraca sumę wszystkich elementów wiersza

Oto graf wywołań dla tej funkcji:



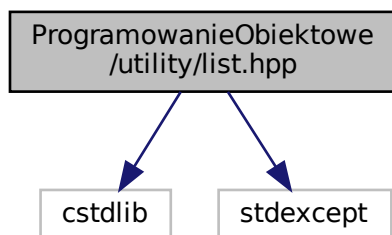
Oto graf wywoływań tej funkcji:



## 5.25 Dokumentacja pliku ProgramowanieObiektowe/utility/list.hpp

```
#include <cstdlib>
#include <stdexcept>
```

Wykres zależności załączania dla list.hpp:



### Komponenty

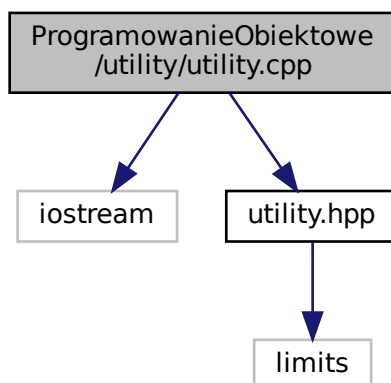
- class [List< T >](#)

The [List](#) Własna implementacja klasy `List/Vector`.

## 5.26 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.*

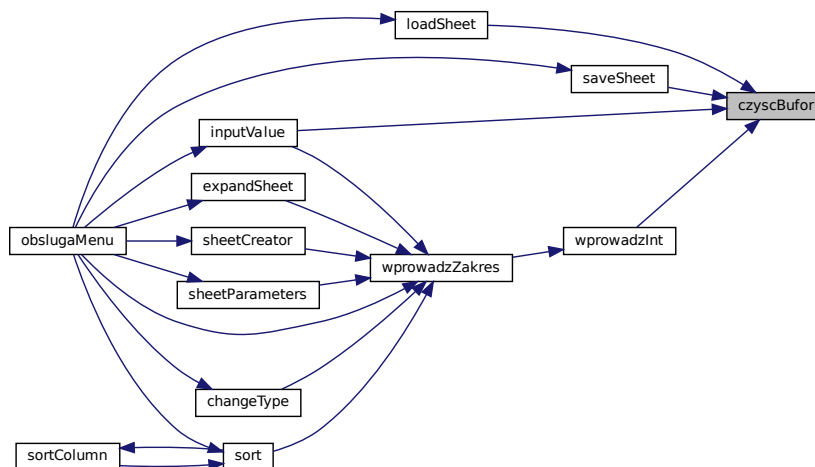
### 5.26.1 Dokumentacja funkcji

#### 5.26.1.1 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:



### 5.26.1.2 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.

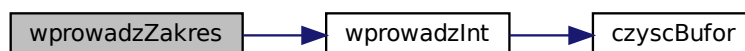
#### Parametry

in	min	Minimalna wartość jaką można wprowadzić
in	max	Maksymalna wartość jaką można wprowadzić

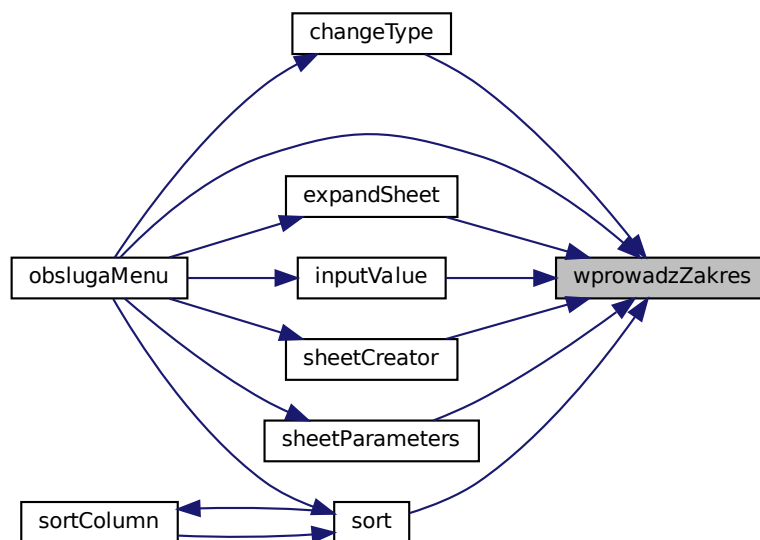
#### Zwraca

Wartość z zakresu <min; max>

Oto graf wywołań dla tej funkcji:



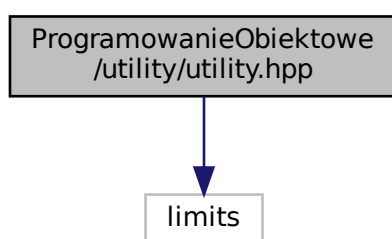
Oto graf wywoływań tej funkcji:



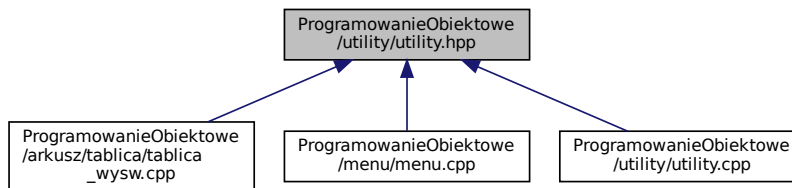
## 5.27 Dokumentacja pliku ProgramowanieObiektowe/utility/utility.hpp

```
#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 `czyszcBufor()`  
*Funkcja od czyszczenia buforu strumienia CIN.*

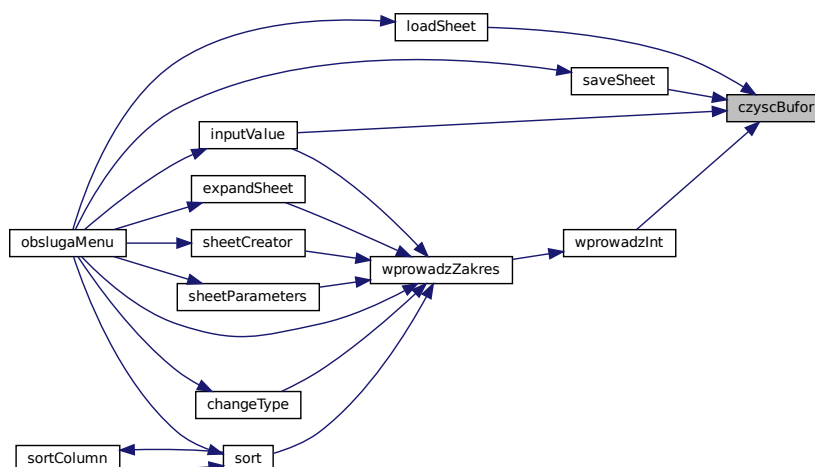
### 5.27.1 Dokumentacja funkcji

#### 5.27.1.1 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:



### 5.27.1.2 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.

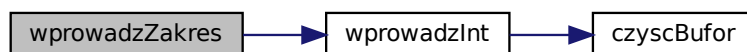
#### Parametry

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

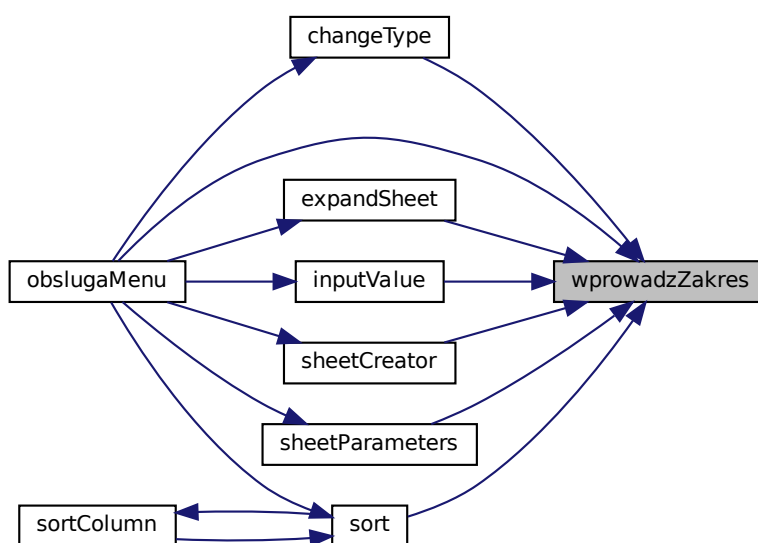
#### Zwraca

Wartość z zakresu <min; max>

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



# Index

BadFileException, 7  
begin  
    Column, 14  
    List< T >, 26  
Cell, 8  
    operator<, 11  
    operator>, 11  
    operator+, 10  
CellType  
    cellType.hpp, 56  
cellType.hpp  
    CellType, 56  
    IntCell, 56  
changeType  
    menu.cpp, 82  
    menu.hpp, 93  
Column, 12  
    begin, 14  
    Column, 13  
    end, 14  
    generateCellArray, 14  
    getCell, 15  
    getHeight, 16  
    getType, 16  
    max, 16  
    min, 16  
    operator[], 17  
    resize, 17  
    sum, 18  
columnParameters  
    menu.cpp, 82  
    menu.hpp, 94  
columnWidth  
    tablica\_wysw.cpp, 67  
    tablica\_wysw.hpp, 70  
countCalculateableColumns  
    operacje.cpp, 103  
    operacje.hpp, 109  
createColumnArray  
    Sheet, 38  
czyscBufor  
    utility.cpp, 114  
    utility.hpp, 117  
Display, 19  
    simpleControl, 19  
    simpleMenu, 20  
DisplaySheet  
    tablica\_wysw.cpp, 68  
    tablica\_wysw.hpp, 71  
end  
    Column, 14  
    List< T >, 26  
error.hpp  
    PLIK\_ACCESS, 74  
    PLIK\_FORMAT, 74  
    PLIK\_ROZMIAR, 74  
    TABLICA\_SIZE, 74  
    TABLICA\_ZAKR, 74  
    Wyjatk, 74  
expandSheet  
    menu.cpp, 83  
    menu.hpp, 94  
fileOperations.cpp  
    loadFile, 75  
    saveFile, 76  
fileOperations.hpp  
    loadFile, 78  
    saveFile, 79  
generateCellArray  
    Column, 14  
generujMenu  
    menu.cpp, 84  
    menu.hpp, 95  
getCalcValue  
    IntCell, 24  
    NumericCell, 34  
getCell  
    Column, 15  
getColumn  
    Sheet, 39  
getElement  
    List< T >, 27  
getHeight  
    Column, 16  
    Sheet, 40  
getSize  
    List< T >, 27  
getType  
    Column, 16  
getValue  
    IntCell, 24  
    StringCell, 46  
    StringInterface, 52  
getWidth  
    Sheet, 40

- inputValue
  - menu.cpp, 84
  - menu.hpp, 95
- IntCell, 21
  - cellType.hpp, 56
  - getCalcValue, 24
  - getValue, 24
  - IntCell, 23
  - setValue, 24
- List
  - List< T >, 26
- List< T >, 25
  - begin, 26
  - end, 26
  - getElement, 27
  - getSize, 27
  - List, 26
  - operator[], 28
  - pop, 28
  - push, 30
- loadFile
  - fileOperations.cpp, 75
  - fileOperations.hpp, 78
- loadSheet
  - menu.cpp, 85
  - menu.hpp, 96
- max
  - Column, 16
- maxRow
  - operacje.cpp, 104
  - operacje.hpp, 109
- menu.cpp
  - changeType, 82
  - columnParameters, 82
  - expandSheet, 83
  - generujMenu, 84
  - inputValue, 84
  - loadSheet, 85
  - obslugaMenu, 86
  - rowParameters, 87
  - saveSheet, 88
  - sheetCreator, 89
  - sheetParameters, 90
  - sort, 91
- menu.hpp
  - changeType, 93
  - columnParameters, 94
  - expandSheet, 94
  - generujMenu, 95
  - inputValue, 95
  - loadSheet, 96
  - obslugaMenu, 97
  - rowParameters, 98
  - saveSheet, 99
  - sheetCreator, 100
  - sheetParameters, 101
  - sort, 102
- min
  - Column, 16
- minRow
  - operacje.cpp, 105
  - operacje.hpp, 110
- NotNumericValue, 30
- NumericCell, 32
  - getCalcValue, 34
  - operator<, 35
  - operator>, 35
  - operator+, 34
- obslugaMenu
  - menu.cpp, 86
  - menu.hpp, 97
- operacje.cpp
  - countCalculateableColumns, 103
  - maxRow, 104
  - minRow, 105
  - sortColumn, 106
  - sumRow, 107
- operacje.hpp
  - countCalculateableColumns, 109
  - maxRow, 109
  - minRow, 110
  - sortColumn, 111
  - sumRow, 112
- operator<
  - Cell, 11
  - NumericCell, 35
  - StringCell, 47, 48
- operator>
  - Cell, 11
  - NumericCell, 35
  - StringCell, 48, 49
- operator+
  - Cell, 10
  - NumericCell, 34
  - StringCell, 47
- operator[]
  - Column, 17
  - List< T >, 28
  - Sheet, 41
- PLIK\_ACCESS
  - error.hpp, 74
- PLIK\_FORMAT
  - error.hpp, 74
- PLIK\_ROZMIAR
  - error.hpp, 74
- pop
  - List< T >, 28
- ProgramowanieObiektowe/arkusz/cell/cell.hpp, 55
- ProgramowanieObiektowe/arkusz/cell/cellType.hpp, 56
- ProgramowanieObiektowe/arkusz/cell/intCell.cpp, 57
- ProgramowanieObiektowe/arkusz/cell/intCell.hpp, 57
- ProgramowanieObiektowe/arkusz/cell/numericCell.cpp, 59



- ProgramowanieObiektowe/arkusz/cell/numericCell.hpp, 59
- ProgramowanieObiektowe/arkusz/cell/stringCell.cpp, 61
- ProgramowanieObiektowe/arkusz/cell/stringCell.hpp, 61
- ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp, 62
- ProgramowanieObiektowe/arkusz/column/column.cpp, 63
- ProgramowanieObiektowe/arkusz/column/column.hpp, 64
- ProgramowanieObiektowe/arkusz/tablica/tablica.cpp, 65
- ProgramowanieObiektowe/arkusz/tablica/tablica.hpp, 66
- ProgramowanieObiektowe/arkusz/tablica/tablica\_wysw.cpp, 66
- ProgramowanieObiektowe/arkusz/tablica/tablica\_wysw.hpp, 69
- ProgramowanieObiektowe/CursesUI/display.hpp, 72
- ProgramowanieObiektowe/error.hpp, 73
- ProgramowanieObiektowe/io/fileOperations.cpp, 74
- ProgramowanieObiektowe/io/fileOperations.hpp, 77
- ProgramowanieObiektowe/main.cpp, 80
- ProgramowanieObiektowe/menu/menu.cpp, 81
- ProgramowanieObiektowe/menu/menu.hpp, 92
- ProgramowanieObiektowe/operacje/operacje.cpp, 102
- ProgramowanieObiektowe/operacje/operacje.hpp, 108
- ProgramowanieObiektowe/utility/list.hpp, 113
- ProgramowanieObiektowe/utility/utility.cpp, 114
- ProgramowanieObiektowe/utility/utility.hpp, 116
- push
  - List< T >, 30
- resize
  - Column, 17
  - Sheet, 42
- rowParameters
  - menu.cpp, 87
  - menu.hpp, 98
- saveFile
  - fileOperations.cpp, 76
  - fileOperations.hpp, 79
- saveSheet
  - menu.cpp, 88
  - menu.hpp, 99
- setValue
  - IntCell, 24
  - StringCell, 49
  - StringInterface, 52
- Sheet, 36
  - createColumnArray, 38
  - getColumn, 39
  - getHeight, 40
  - getWidth, 40
  - operator[], 41
  - resize, 42
  - Sheet, 37
- sheetCreator
  - menu.cpp, 89
- menu.hpp, 100
- sheetParameters
  - menu.cpp, 90
  - menu.hpp, 101
- simpleControl
  - Display, 19
- simpleMenu
  - Display, 20
- sort
  - menu.cpp, 91
  - menu.hpp, 102
- sortColumn
  - operacje.cpp, 106
  - operacje.hpp, 111
- StringCell, 43
  - getValue, 46
  - operator<, 47, 48
  - operator>, 48, 49
  - operator+, 47
  - setValue, 49
- StringInterface, 50
  - getValue, 52
  - setValue, 52
- sum
  - Column, 18
- sumRow
  - operacje.cpp, 107
  - operacje.hpp, 112
- TABLICA\_SIZE
  - error.hpp, 74
- tablica\_wysw.cpp
  - columnWidth, 67
  - DisplaySheet, 68
- tablica\_wysw.hpp
  - columnWidth, 70
  - DisplaySheet, 71
- TABLICA\_ZAKR
  - error.hpp, 74
- utility.cpp
  - czyscBufor, 114
  - wprowadzZakres, 115
- utility.hpp
  - czyscBufor, 117
  - wprowadzZakres, 117
- wprowadzZakres
  - utility.cpp, 115
  - utility.hpp, 117
- Wyjatki
  - error.hpp, 74