Arkusz Kalkulacyjny

Kamil Czop
Projekt Programowanie Obiektowe

Chapter 1

Indeks hierarchiczny

1.1 Hierarchia klas

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

Chapter 2

Indeks klas

2.1 Lista klas

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

Chapter 3

Indeks plików

3.1 Lista plików

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

Chapter 4

Dokumentacja klas

4.1 Dokumentacja struktury BadFileException

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

`#include <error.hpp>`

Diagram dziedziczenia dla BadFileException

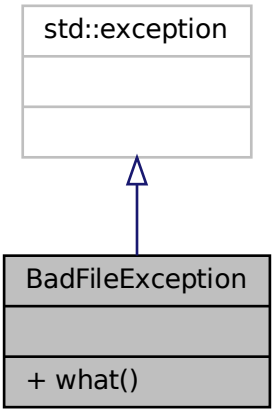
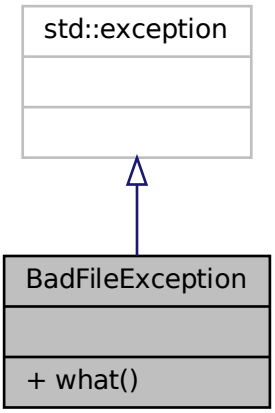


Diagram współpracy dla BadFileException:



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 dostęplu do pliku lub nieistnieje (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

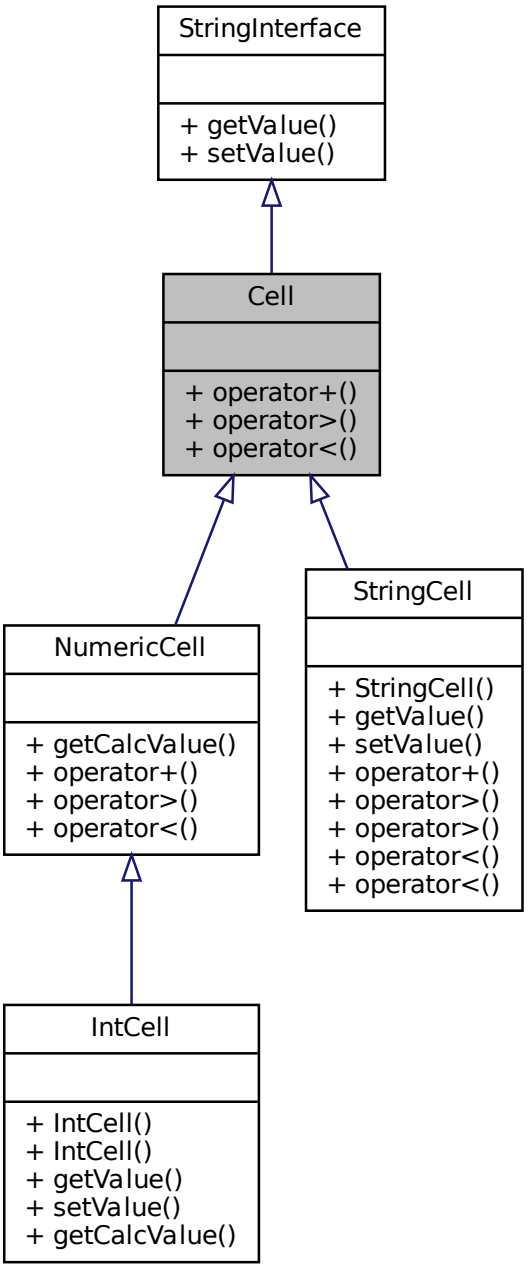
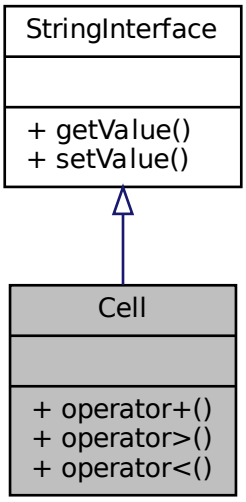


Diagram współpracy dla Cell:



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 róznić.
- 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 róznić.

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

in	*rhs*	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 róznić.

Parametry

rhs	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 róznić.

Parametry

rhs	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:

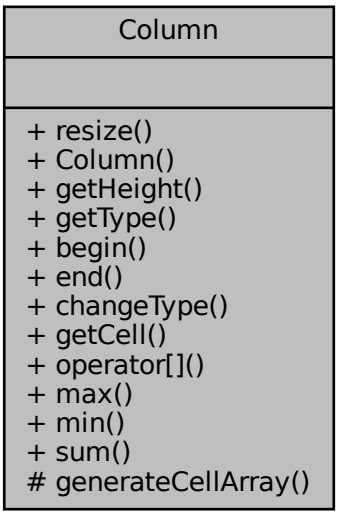
- ProgramowanieObiektowe/arkusz/cell/cell.hpp

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 zwracaja 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.

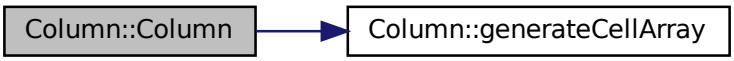
Parametry

in	*height*	Rozmiar tablicy komórek w kolumnie
in	*type*	Typ tworzonych komórek w kolumnie

Wyjątki

std::bad_array_new_length	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 zwracaja 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	*height*	Wysokość nowej tablicy
in	*type*	Typ tworzonych komórek

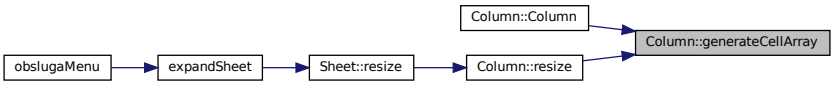
Wyjątki

std::bad_array_new_length	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	*y*	współżędna do komórki w tablicy

Wyjątki

std::out_of_range	Gdy y jest poza zakresem kolumny (y $>$ 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 przechowywuje

Zwraca

Typ komórek w kolumnie

4.3.3.7 max()

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

Funkcja szukania maksymalnej wartości kolumny.

Funkcja przeszukuje całą kolumne 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 kolumne 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*	współżędna do komórki w tablicy

Wyjątki

std::out_of_range	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

std::bad_array_new_length	w przypadku zerowego rozmiaru

Parametry

in	*newHeight*	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:

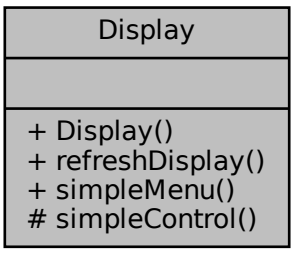
- ProgramowanieObiektowe/arkusz/column/column.hpp
- ProgramowanieObiektowe/arkusz/column/column.cpp

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()

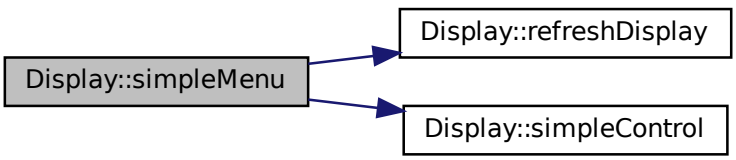
```
int Display::simpleMenu ( )
```

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

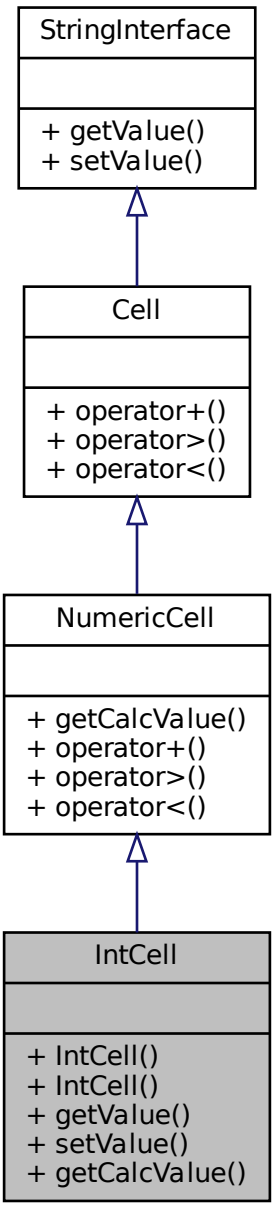
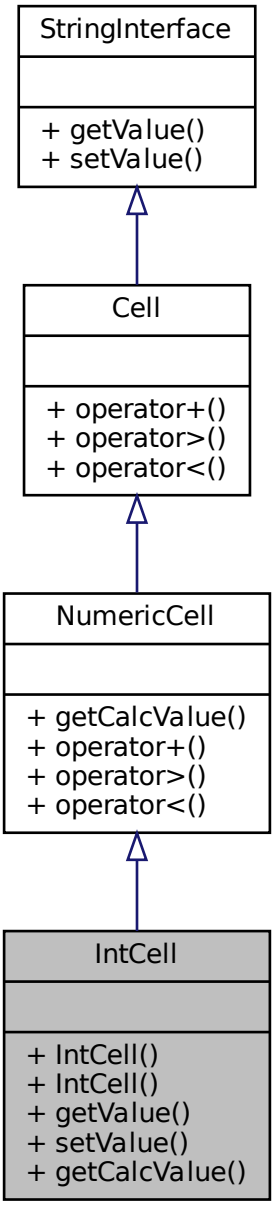


Diagram współpracy dla IntCell:



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

NotNumericValue	Brak możliwości przetworzenia wartości tekstowej na liczbową

Parametry

in	*value*	ustawiana wartość

Implementuje StringInterface.

Dokumentacja dla tej klasy została wygenerowana z plików:

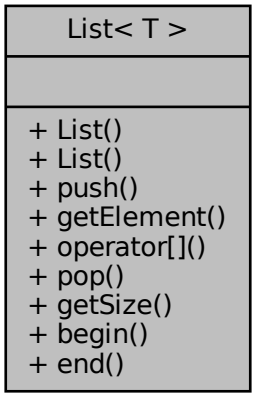
- ProgramowanieObiektowe/arkusz/cell/intCell.hpp
- ProgramowanieObiektowe/arkusz/cell/intCell.cpp

4.6 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	*elements*	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	*index*	Indeks elementu w tablicy listy

Wyjątki

std::out_of_range	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	*index*	Indeks elementu w tablicy listy

Wyjątki

std::out_of_range	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

std::bad_array_new_length	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	*element*	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

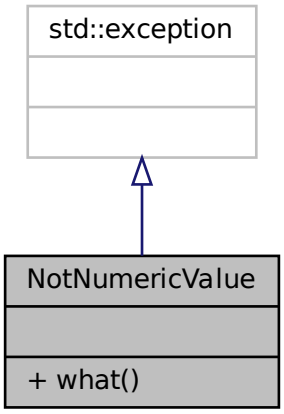
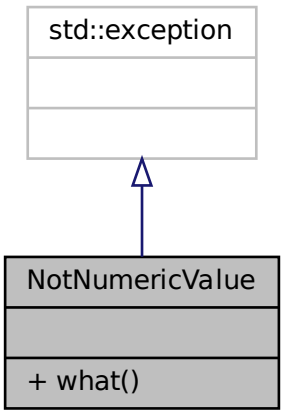


Diagram współpracy dla NotNumericValue:



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

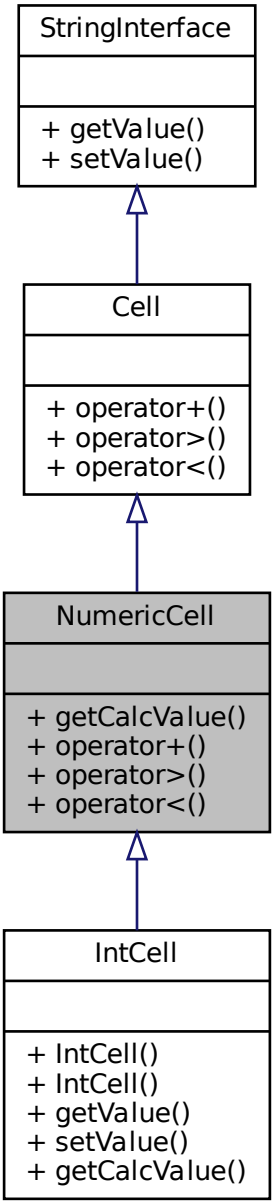
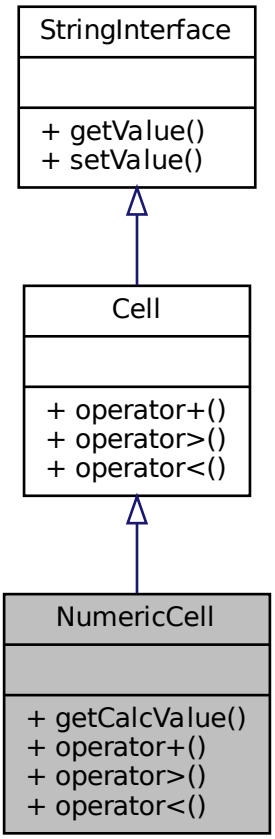


Diagram współpracy dla NumericCell:



Metody publiczne

- virtual double getCalcValue ()=0

  getCalcValue Metoda klasy umożliwiająca bezpośrednie uzyskanie wartości double Metoda umożliwia uzyskanie wartości bezpośrednie 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]
```

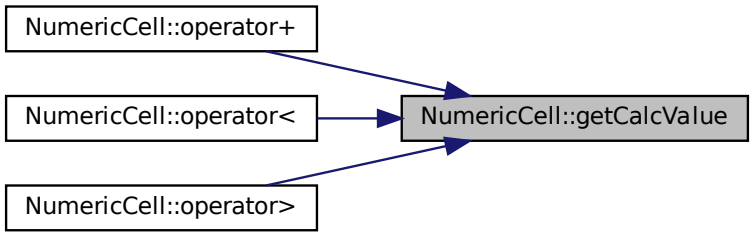
getCalcValue Metoda klasy umożliwiająca bezpośrednie uzyskanie wartości double Metoda umożliwia uzyskanie wartości bezpośrednie bez potrzeby parsowania wartości

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

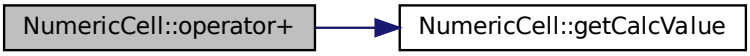
in	*rhs*	wartość którą sumujemy z komórką

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

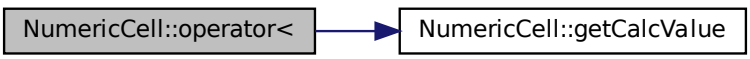
rhs	komórka którą porównujemy

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

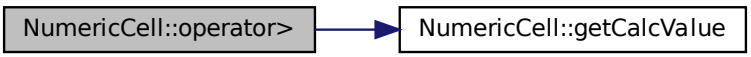
rhs	komórka którą porównujemy

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:

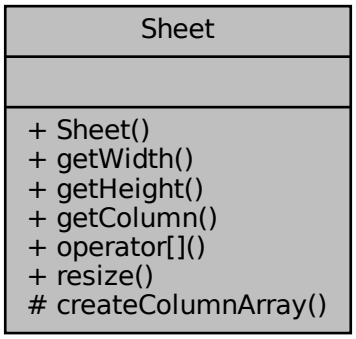
- ProgramowanieObiektowe/arkusz/cell/numericCell.hpp
- ProgramowanieObiektowe/arkusz/cell/numericCell.cpp

4.9 Dokumentacja klasy Sheet

Klasa opisująca Arkusz Klasa Arkusz przechowywujaca 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 akrusz 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 przechowywujaca tablicę kolumn i jej rozmiar.

4.9.2 Dokumentacja konstruktora i destruktora

4.9.2.1 Sheet()

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

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

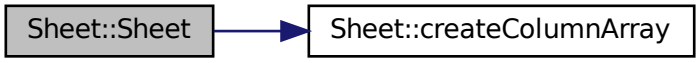
Parametry

in	*width*	Szerokość tablicy nowego arkusza
in	*height*	Wysokość nowej tablicy
in	*types*	Typy tworzonych kolumn

Wyjątki

bad_array_new_length	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 tablice. Funkcja generujaca tablicę o określonym rozmiarze

Parametry

in	*width*	Szerokość nowej tablicy - ilość kolumn
in	*height*	Wysokość nowej tablicy - ilość komórek w kolumnach
in	*types*	Typy tworzonych kolumn

Wyjątki

bad_array_new_length	Zły rozmiar tworzonego arkusza

Zwraca

Tworzy nową tabilce 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

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

Zwraca

Referencja na kolumnę

Oto graf wywoływań tej funkcji:

4.9.3.3 getHeight()

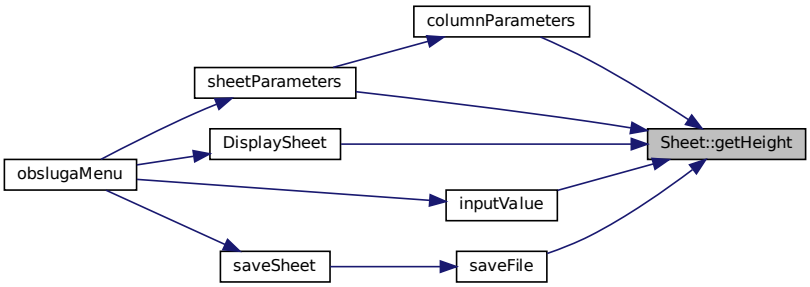
```
size_t Sheet::getHeight ( )
```

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()

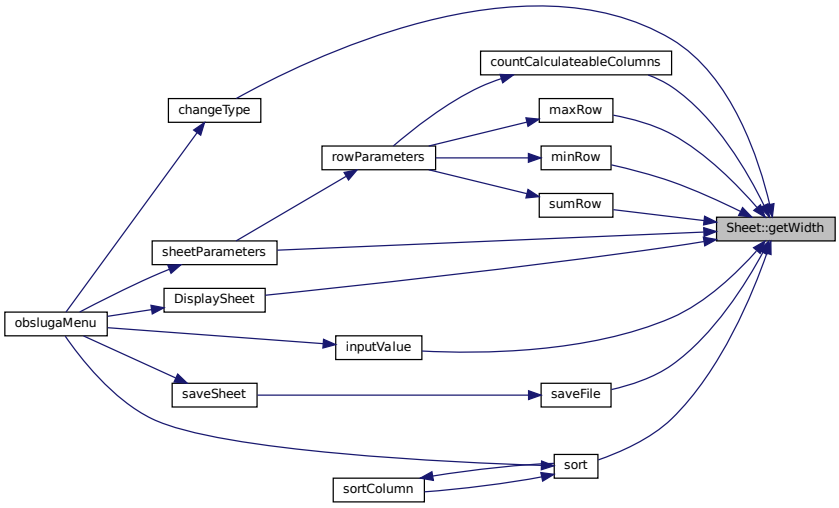
```
size_t Sheet::getWidth ( )
```

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

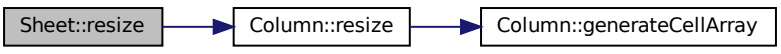
Parametry

in	*x*	Nowa szerokość arkusza
in	*y*	Nowa wysokość kolumn arkusza

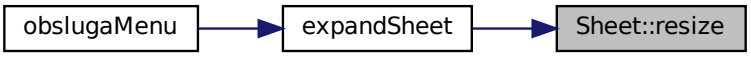
Wyjątki

bad_array_new_length	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

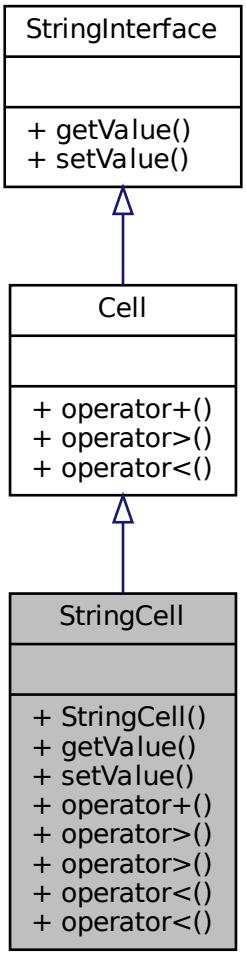
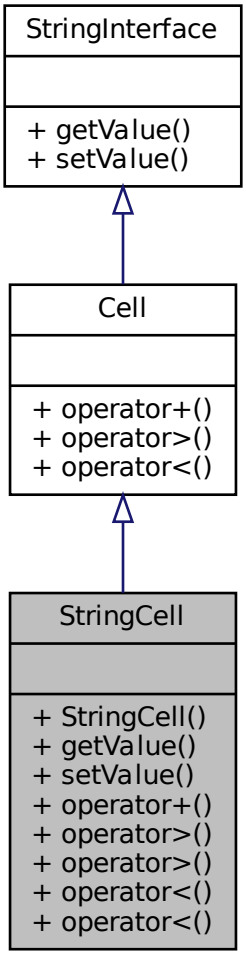


Diagram współpracy dla StringCell:



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]
```

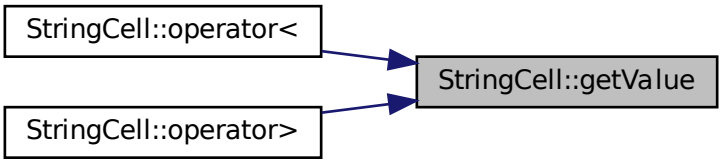
getValue Metoda ogólna od uzyskiwania wartości komórki Metoda zwraca wartość w postaci wartości tekstowej komórki

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

rhs	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

rhs	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

rhs	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	*value*	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

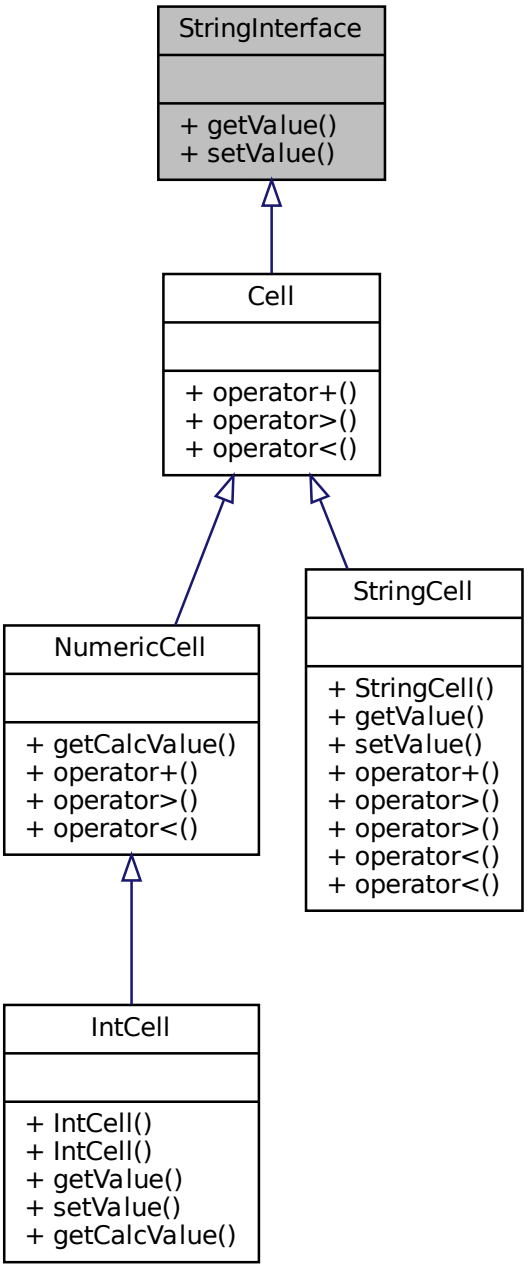
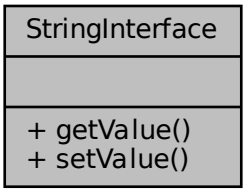


Diagram współpracy dla StringInterface:



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	*value*	ustawiana wartość

Implementowany w StringCell i IntCell.

Dokumentacja dla tej klasy została wygenerowana z pliku:

- ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp

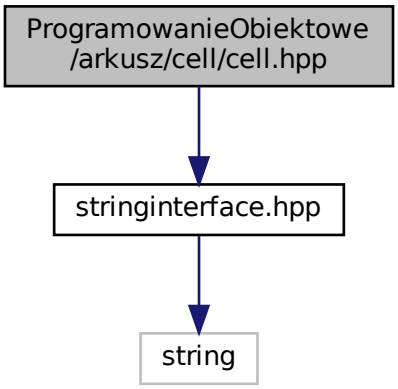
Chapter 5

Dokumentacja plików

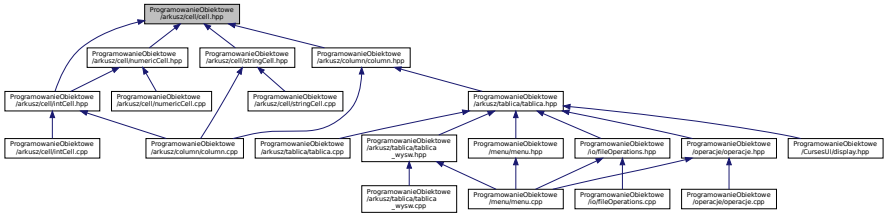
5.1 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/cell.hpp

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

Wykres zależności załączania dla cell.hpp:



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



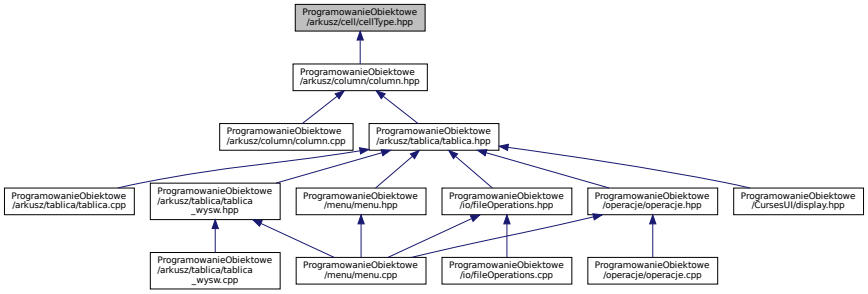
Komponenty

- 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.

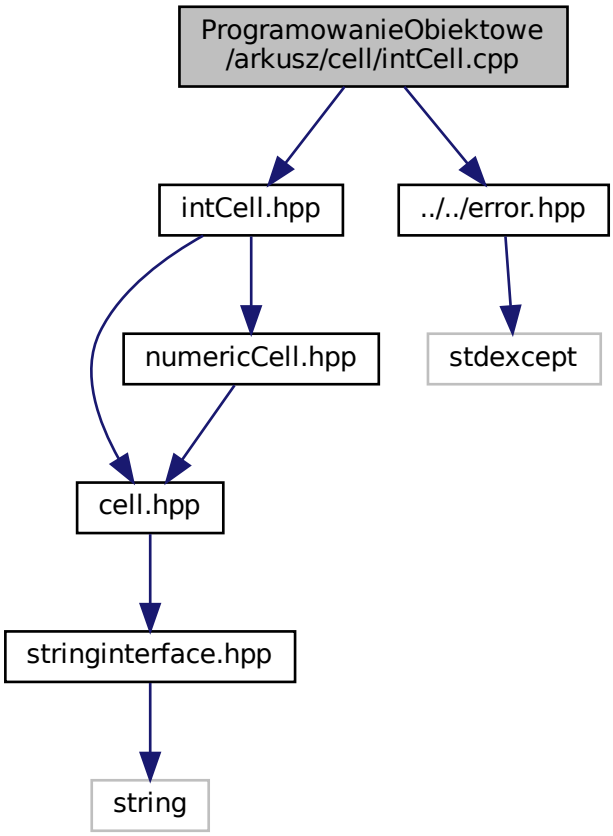
Wartości wyliczeń

IntCell	Komórka tekstowa.

5.3 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/intCell.cpp

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

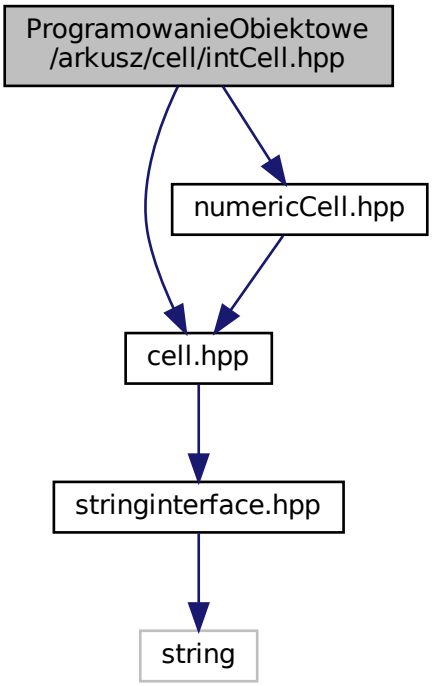
Wykres zależności załączania dla intCell.cpp:



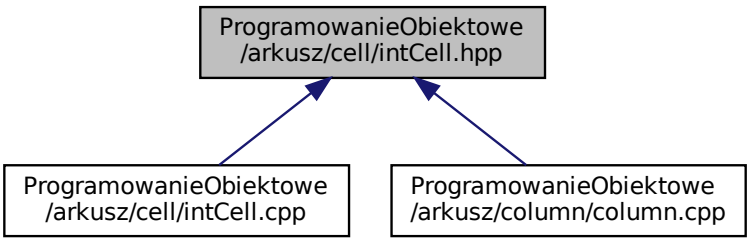
5.4 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/intCell.hpp

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

Wykres zależności załączania dla intCell.hpp:



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



Komponenty

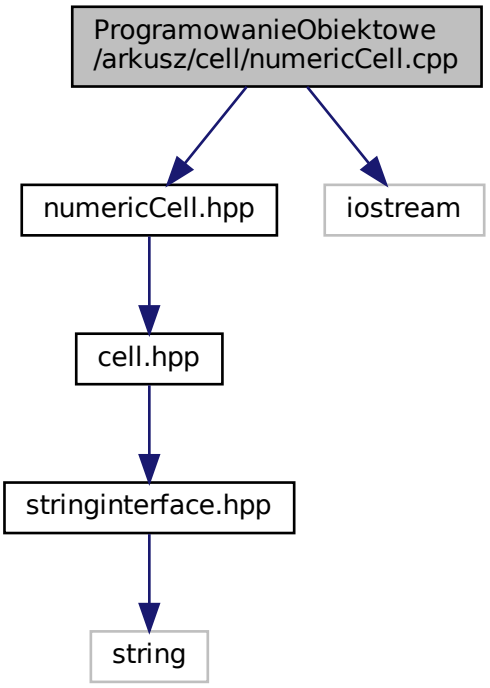
- class IntCell

  IntCell komórka z wartością całkowitą Komórka przyjmująca wartości całkowite.

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

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

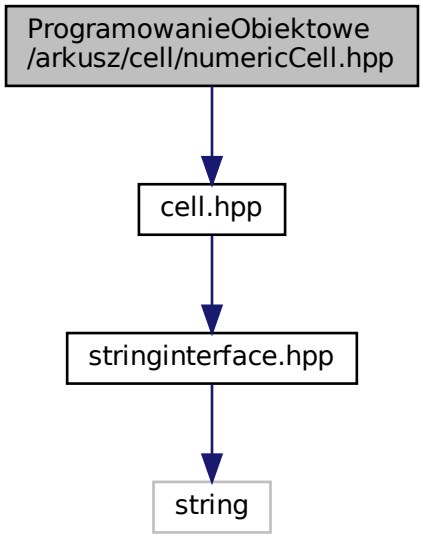
Wykres zależności załączania dla numericCell.cpp:



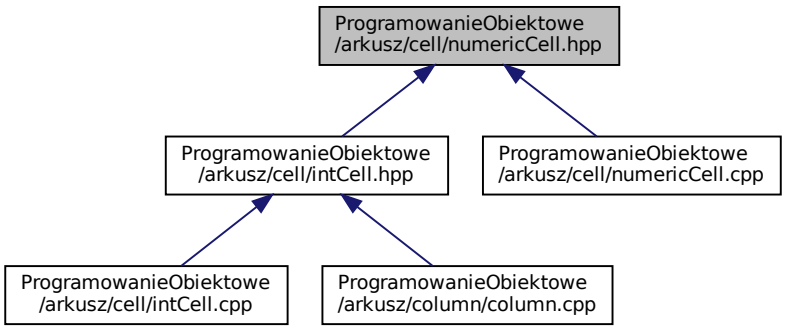
5.6 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/numericCell.hpp

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

Wykres zależności załączania dla numericCell.hpp:



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



Komponenty

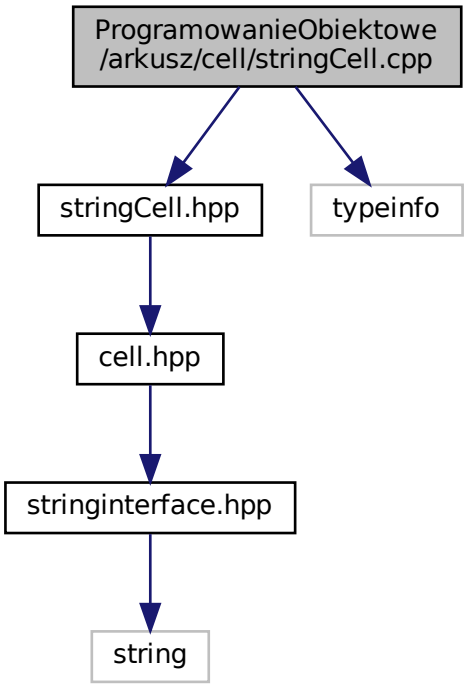
- class NumericCell

  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>
```

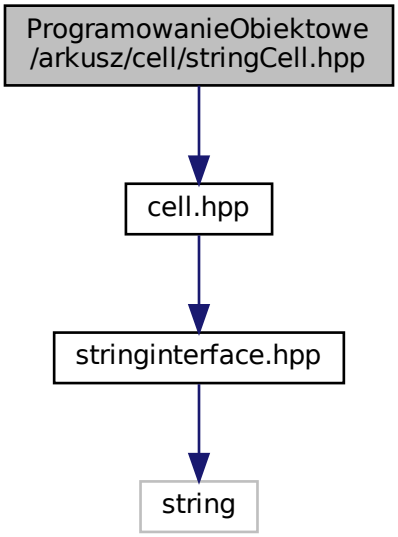
Wykres zależności załączania dla stringCell.cpp:



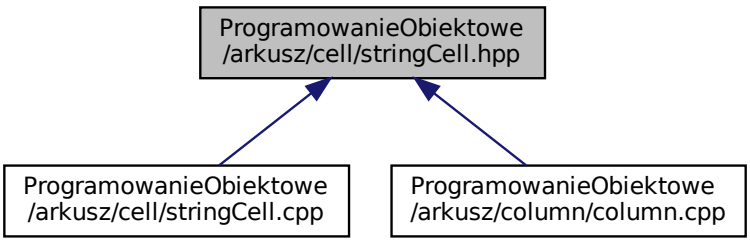
5.8 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/stringCell.hpp

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

Wykres zależności załączania dla stringCell.hpp:



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



Komponenty

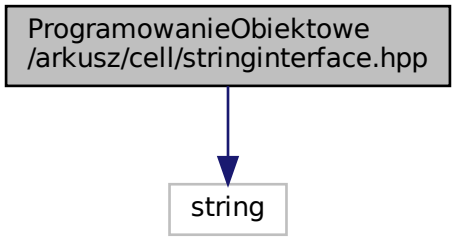
- class StringCell

  StringCell Komórka tekstowa Klasa komórki przyjmującej wartości tekstowe.

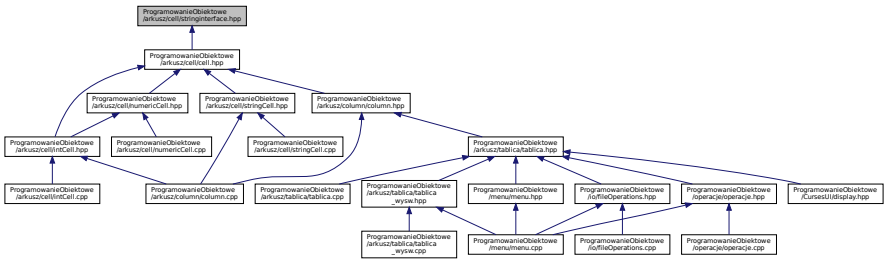
5.9 Dokumentacja pliku ProgramowanieObiektowe/arkusz/cell/stringinterface.hpp

```
#include <string>
```

Wykres zależności załączania dla stringinterface.hpp:



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



Komponenty

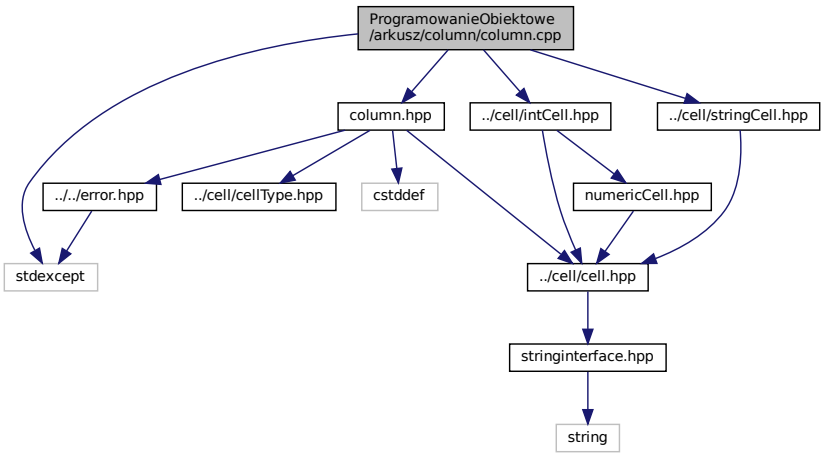
- class StringInterface

  Intefejs elementów przyjmujących/zwracających elementy string.

5.10 Dokumentacja pliku ProgramowanieObiektowe/arkusz/column/column.cpp

```
#include "column.hpp"
#include "../cell/intCell.hpp"
#include "../cell/stringCell.hpp"
#include <stdexcept>
```

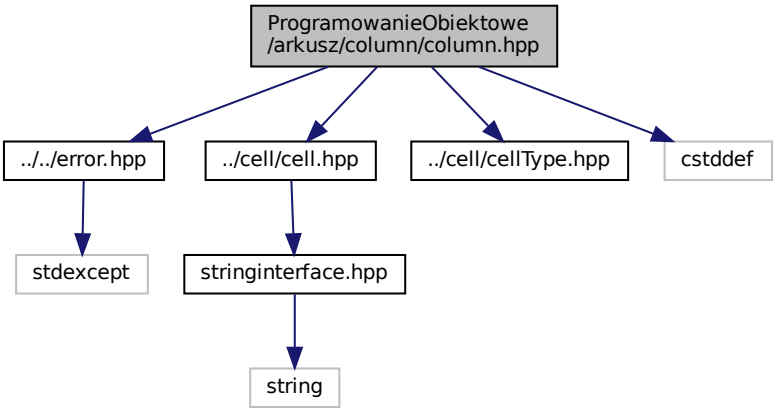
Wykres zależności załączania dla column.cpp:



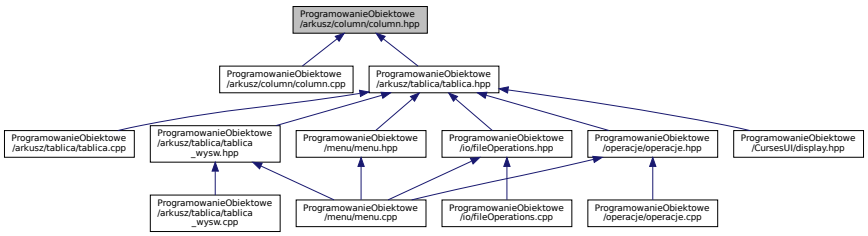
5.11 Dokumentacja pliku ProgramowanieObiektowe/arkusz/column/column.hpp

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

Wykres zależności załączania dla column.hpp:



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



Komponenty

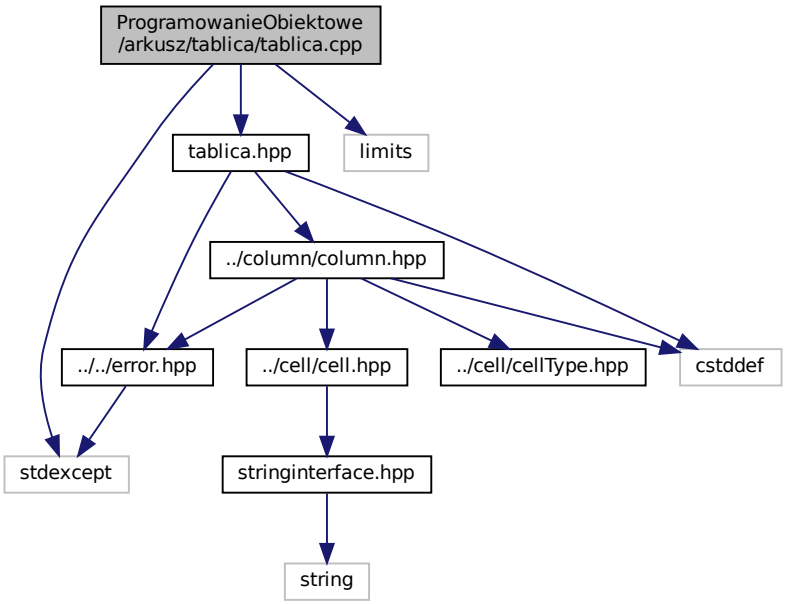
- class Column

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

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

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

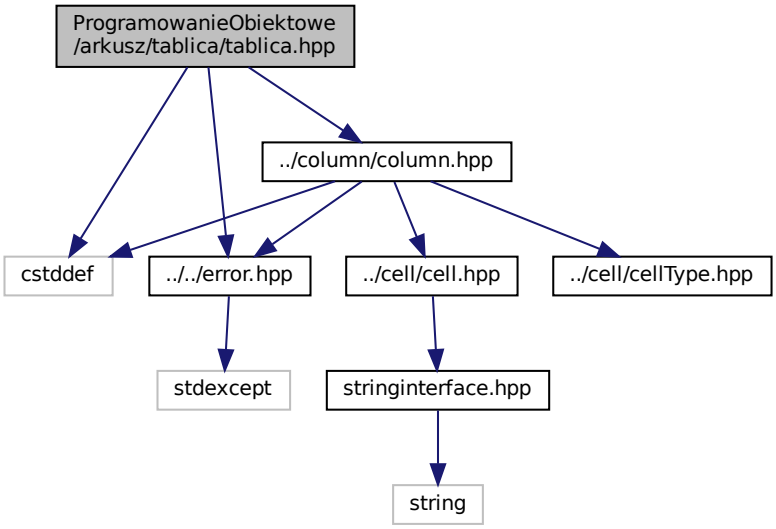
Wykres zależności załączania dla tablica.cpp:



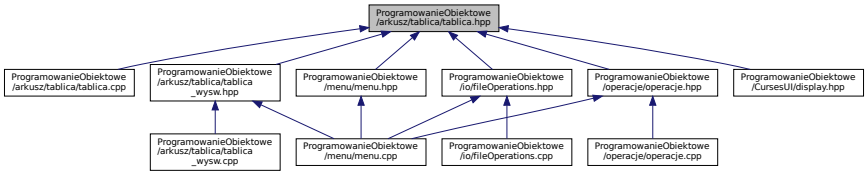
5.13 Dokumentacja pliku ProgramowanieObiektowe/arkusz/tablica/tablica.hpp

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

Wykres zależności załączania dla tablica.hpp:



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



Komponenty

- class Sheet

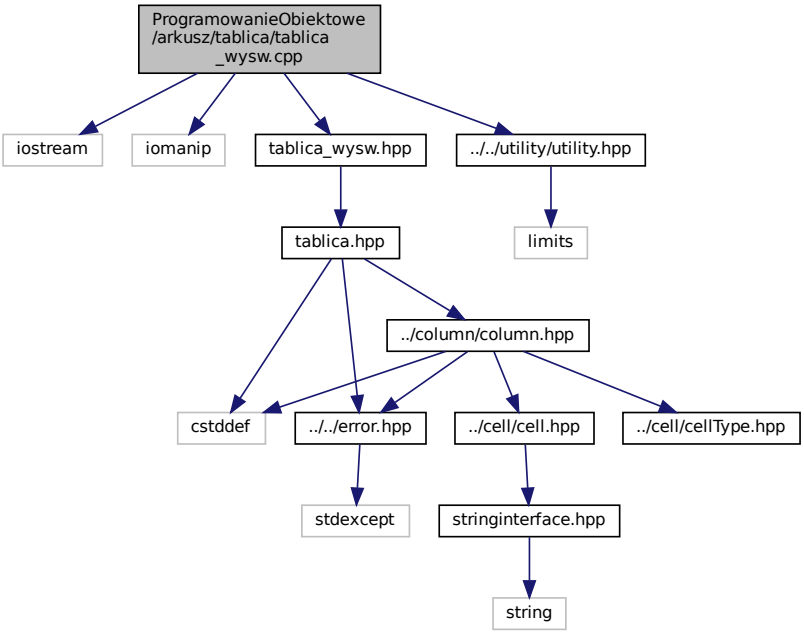
  Klasa opisująca Arkusz Klasa Arkusz przechowywujaca tablicę kolumn i jej rozmiar.

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

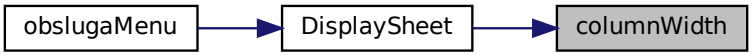
Parametry

in	*column*	Kolumna której długość tekstów będzie sprawdzana

Zwraca

długość najszerszego tekstu

Oto graf wywoływań tej funkcji:



5.14.1.2 DisplaySheet()

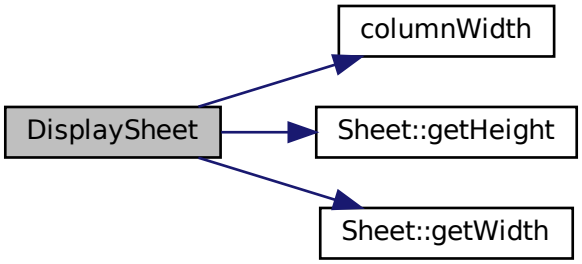
```
void DisplaySheet (
            Sheet sheet )
```

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:

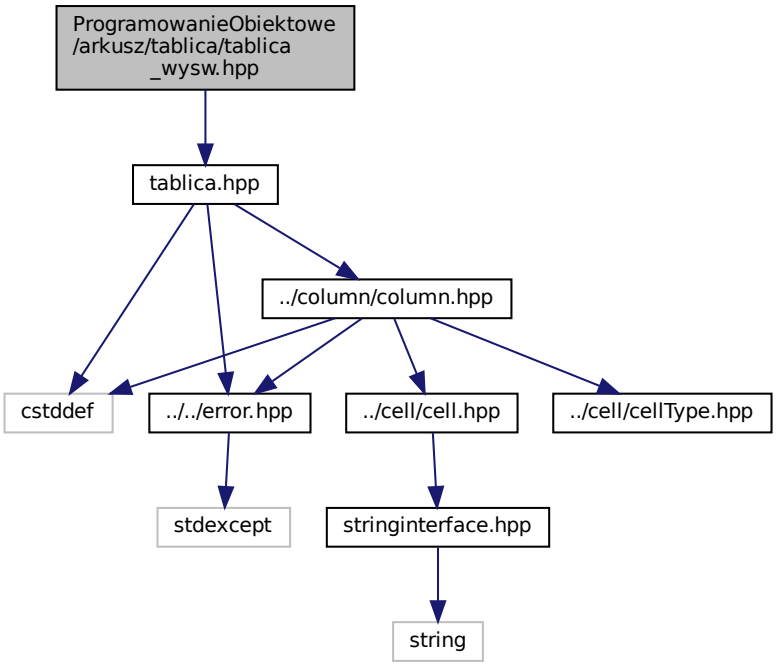


Oto graf wywoływań tej funkcji:

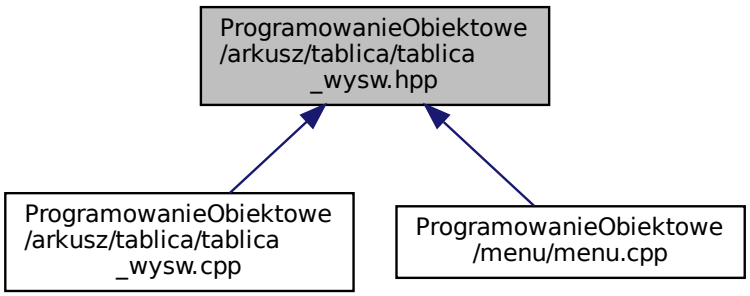
5.15 Dokumentacja pliku ProgramowanieObiektowe/arkusz/tablica/tablica_wysw.hpp

```
#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

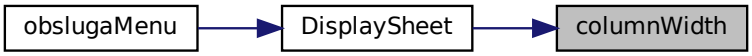
Parametry

in	*column*	Kolumna której długość tekstów będzie sprawdzana

Zwraca

długość najszerszego tekstu

Oto graf wywoływań tej funkcji:



5.15.1.2 DisplaySheet()

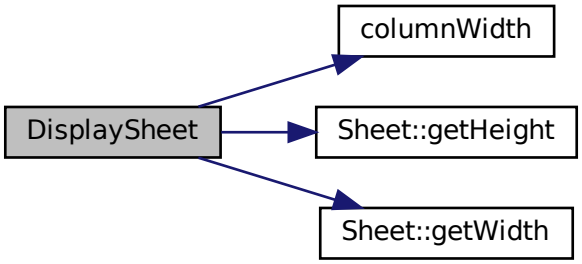
```
void DisplaySheet (
            Sheet sheet )
```

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:

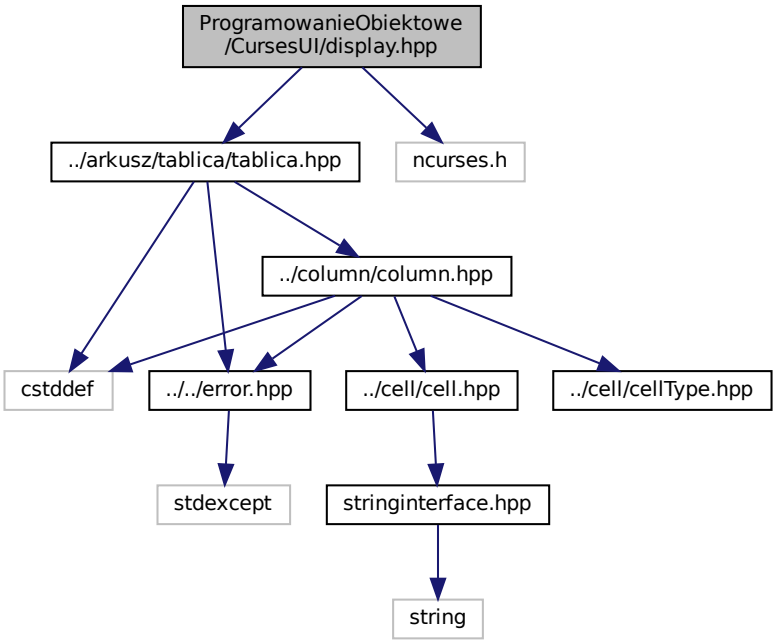


Oto graf wywoływań tej funkcji:

5.16 Dokumentacja pliku ProgramowanieObiektowe/CursesUI/display.hpp

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

Wykres zależności załączania dla display.hpp:



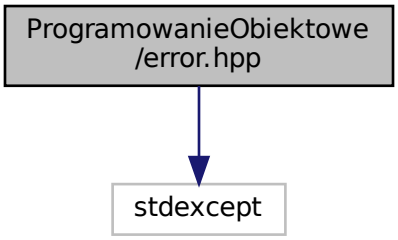
Komponenty

- class Display

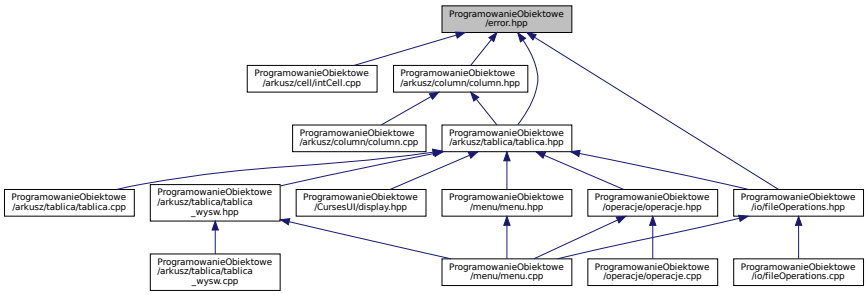
  Display Klasa wyświetlacza Curses.

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 dostęplu do pliku lub nieistnieje (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 Wyjatki

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

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

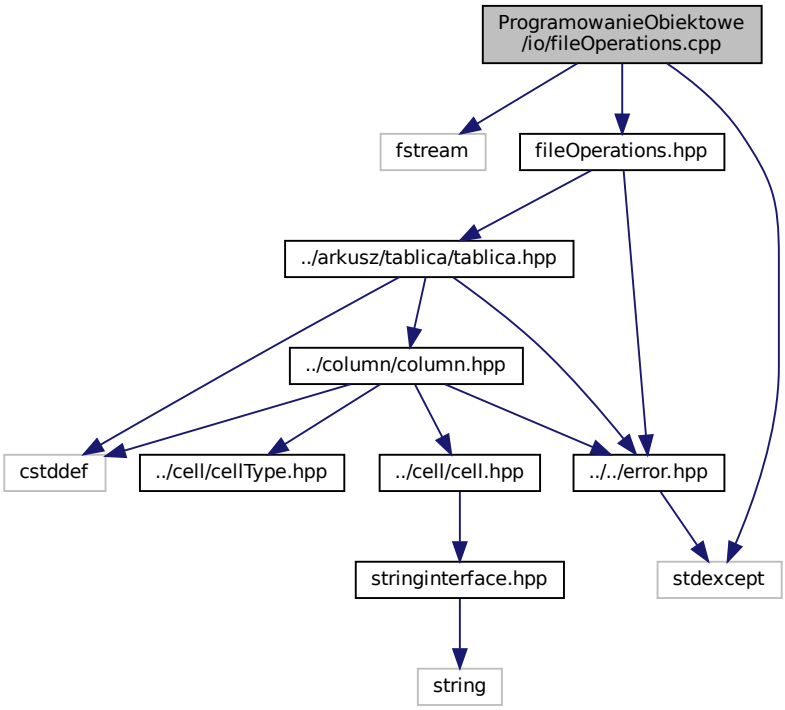
Wartości wyliczeń

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.

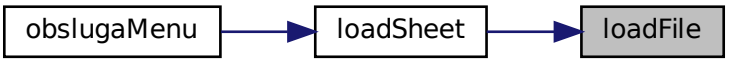
Parametry

in,out	*sheet*	Arkusz do nadpisania wczytywaną tablicą
in	*fileName*	Nazwa wczytywanego pliku

Wyjątki

BadFileException	W przypadku braku pliku lub braku dostępu

Oto graf wywoływań 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

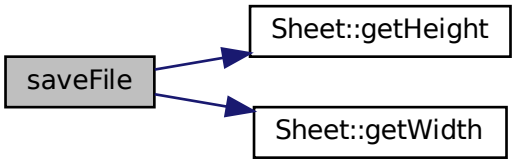
Parametry

in	*sheet*	Arkusz przeznaczony do zapisu
in	*fileName*	Nazwa zapisywanego pliku

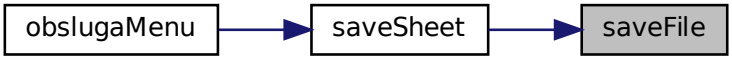
Wyjątki

BadFileException	W przypadku braku dostępu do zapisu

Oto graf wywołań dla tej funkcji:



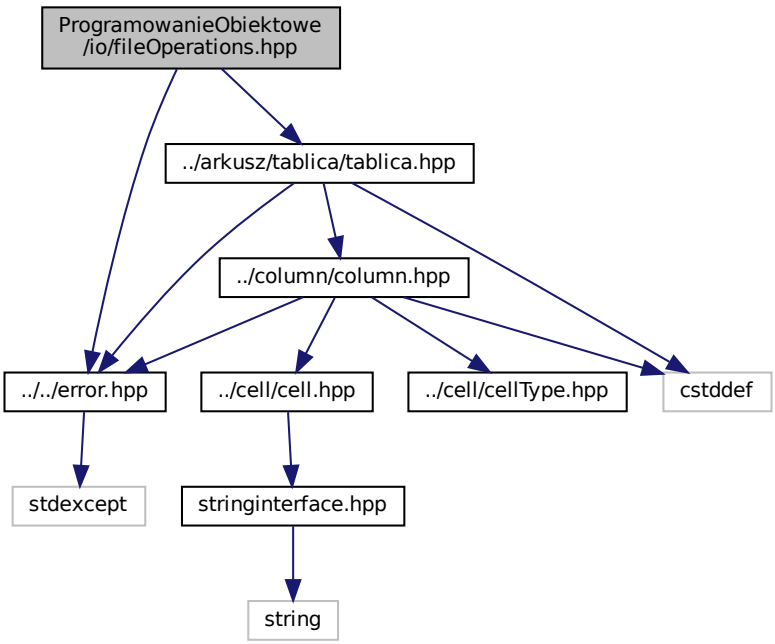
Oto graf wywoływań tej funkcji:



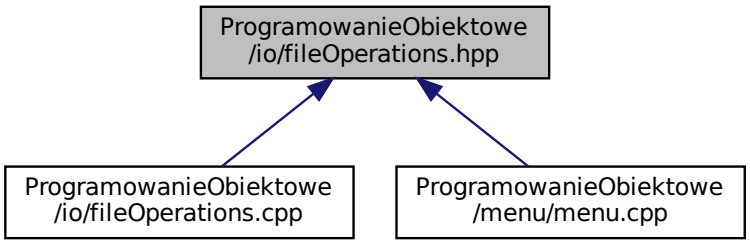
5.19 Dokumentacja pliku ProgramowanieObiektowe/io/fileOperations.hpp

```
#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.

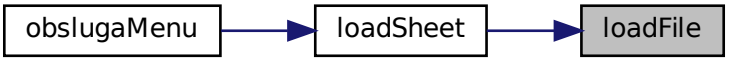
Parametry

in,out	*sheet*	Arkusz do nadpisania wczytywaną tablicą
in	*fileName*	Nazwa wczytywanego pliku

Wyjątki

BadFileException	W przypadku braku pliku lub braku dostępu

Oto graf wywoływań 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

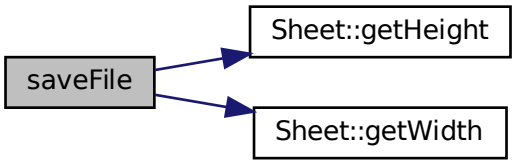
Parametry

in	*sheet*	Arkusz przeznaczony do zapisu
in	*fileName*	Nazwa zapisywanego pliku

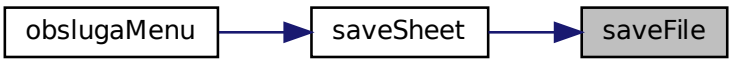
Wyjątki

BadFileException	W przypadku braku dostępu do zapisu

Oto graf wywołań dla tej funkcji:



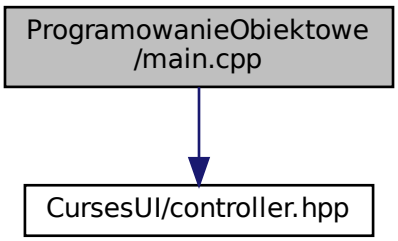
Oto graf wywoływań tej funkcji:



5.20 Dokumentacja pliku ProgramowanieObiektowe/main.cpp

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

Wykres zależności załączania dla main.cpp:



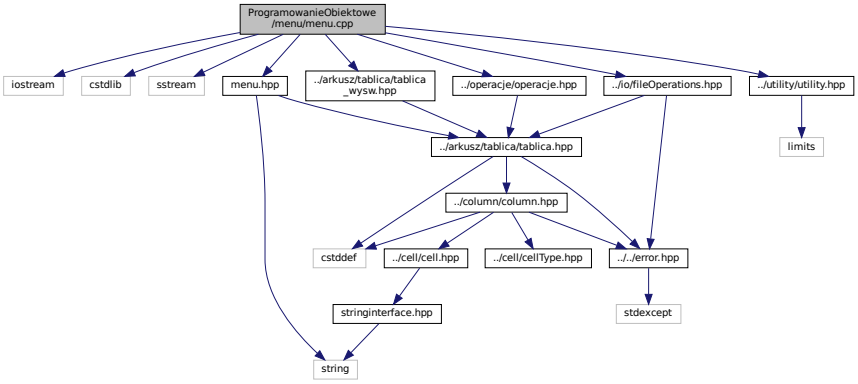
Funkcje

- 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 obslugaMenu ()

  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 warotści który w zależności od typu komórki zmieni jej wartość
- void sheetParameters (Sheet arkusz)

  Funkcja menu od wyboru wgzględem czego wyznaczca 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()

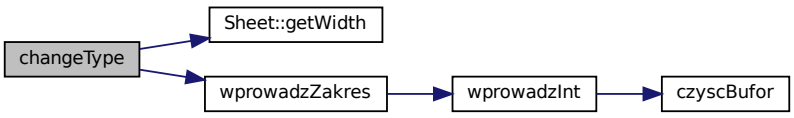
```
void changeType (
            Sheet * arkusz )
```

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.

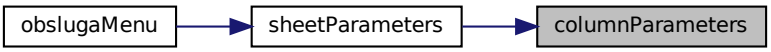
Funkcja zwraca w postaci tekstowej wszystkie parametry wybranej kolumny

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 )
```

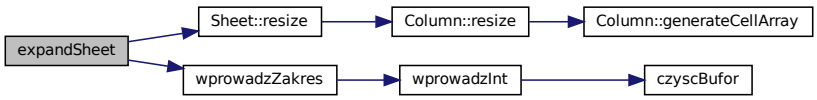
Funkcja modyfikująca rozmiar arkusza.

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()

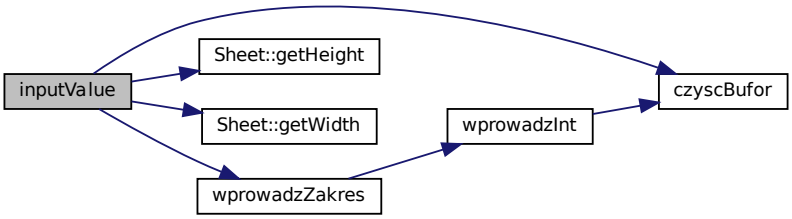
```
void inputValue (
            Sheet * arkusz )
```

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

Parametry

arkusz	Arkusz którego element będzie modyfikowany

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

5.21.1.6 loadSheet()

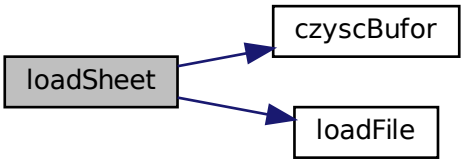
```
void loadSheet (
            Sheet * arkusz )
```

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

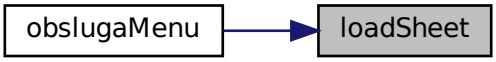
Parametry

in,out	*arkusz*	Arkusz do którego mogą być wczytane elementy

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



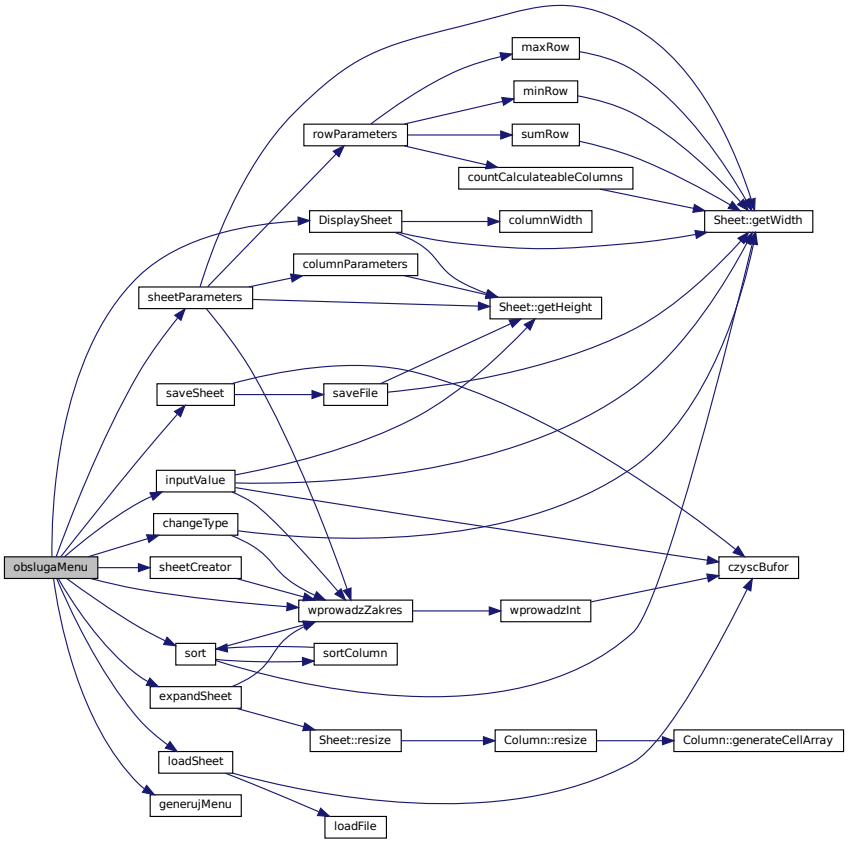
5.21.1.7 obslugaMenu()

```
void obslugaMenu ( )
```

Funkcja kontrolująca działanie programu.

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

funkcji:



5.21.1.8 rowParameters()

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

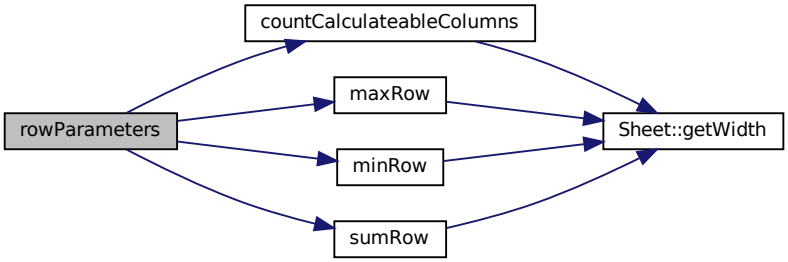
Funkcja od wyznaczania parametrów wiersza arkusza.

Funkcja zwraca w postaci tekstowej wszystkie parametry wybranego wiersza

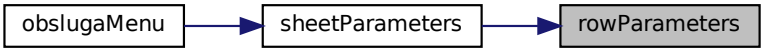
Parametry

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

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.21.1.9 saveSheet()

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

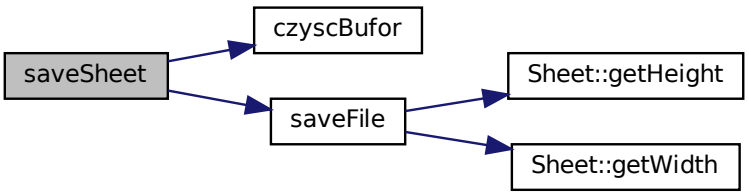
Funkcja menu od zapisu.

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

Parametry

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

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

5.21.1.10 sheetCreator()

```
Sheet sheetCreator ( )
```

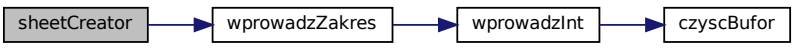
Funkcja tworząca nową tablicę.

Funkcja zawierająca interfejs umożliwiający tworzenie nowego Arkusza z tablicą dwuwymiarową.

Zwraca

Nowy Arkusz do wykorzystywania w programie

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

5.21.1.11 sheetParameters()

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

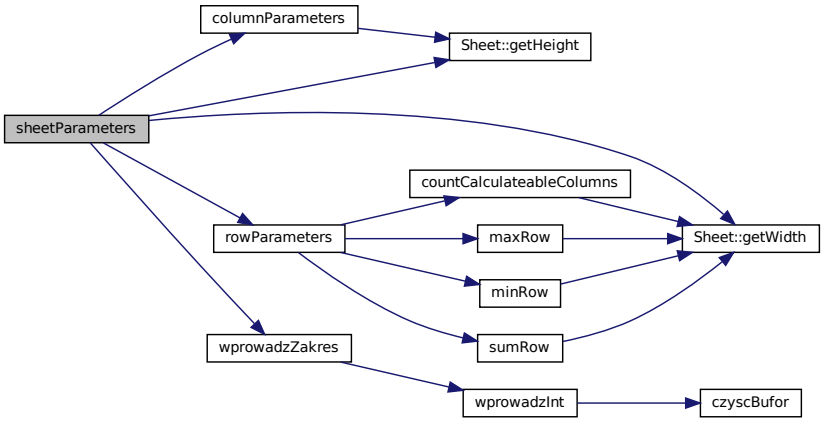
Funkcja menu od wyboru wgzględem czego wyznazcza parametry.

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

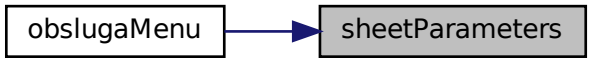
Parametry

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

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



5.21.1.12 sort()

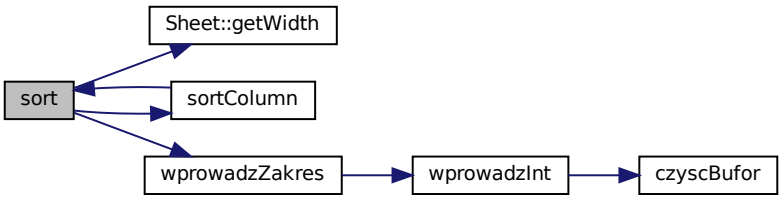
```
void sort (
            Sheet * arkusz )
```

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

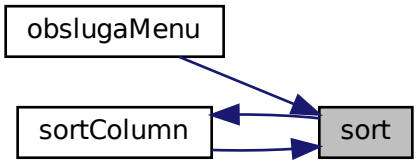
Parametry

in,out	*Arkusz*	którego kolumna będzie sortowana

Oto graf wywołań dla tej funkcji:



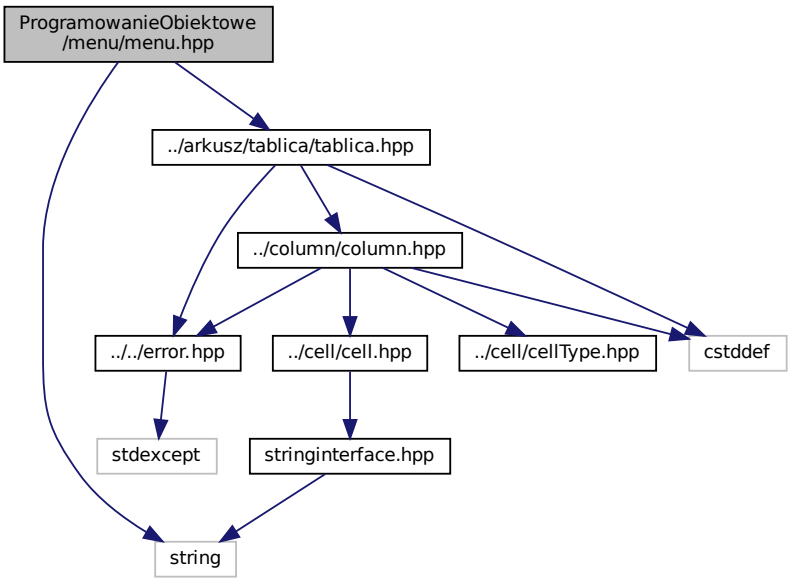
Oto graf wywoływań tej funkcji:



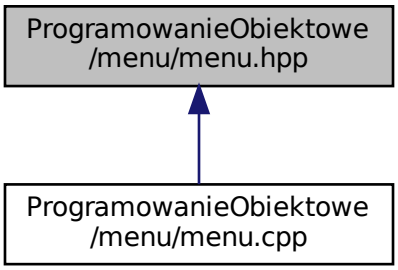
5.22 Dokumentacja pliku ProgramowanieObiektowe/menu/menu.hpp

```
#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 obslugaMenu ()

  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 wgzględem czego wyznazcza 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 warotści 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()

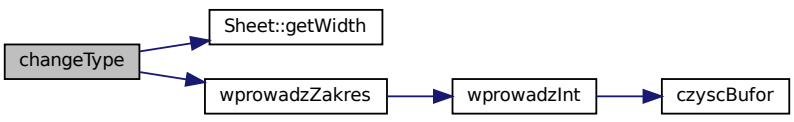
```
void changeType (
            Sheet * arkusz )
```

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	*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:

5.22.1.3 expandSheet()

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

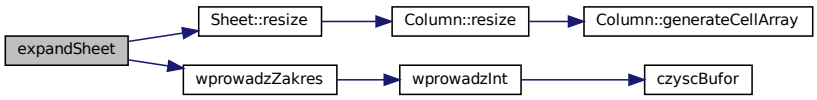
Funkcja modyfikująca rozmiar arkusza.

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()

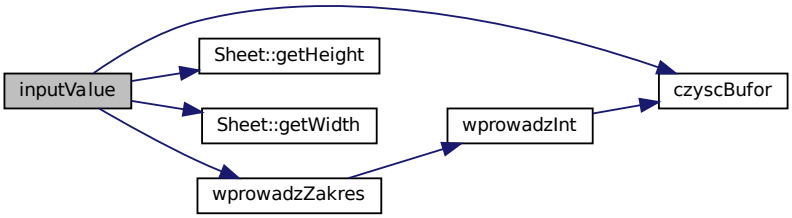
```
void inputValue (
            Sheet * arkusz )
```

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

Parametry

arkusz	Arkusz którego element będzie modyfikowany

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

5.22.1.6 loadSheet()

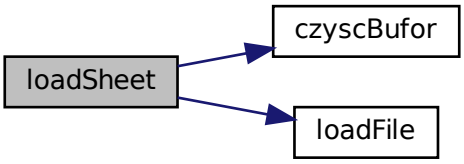
```
void loadSheet (
            Sheet * arkusz )
```

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

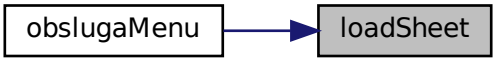
Parametry

in,out	*arkusz*	Arkusz do którego mogą być wczytane elementy

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



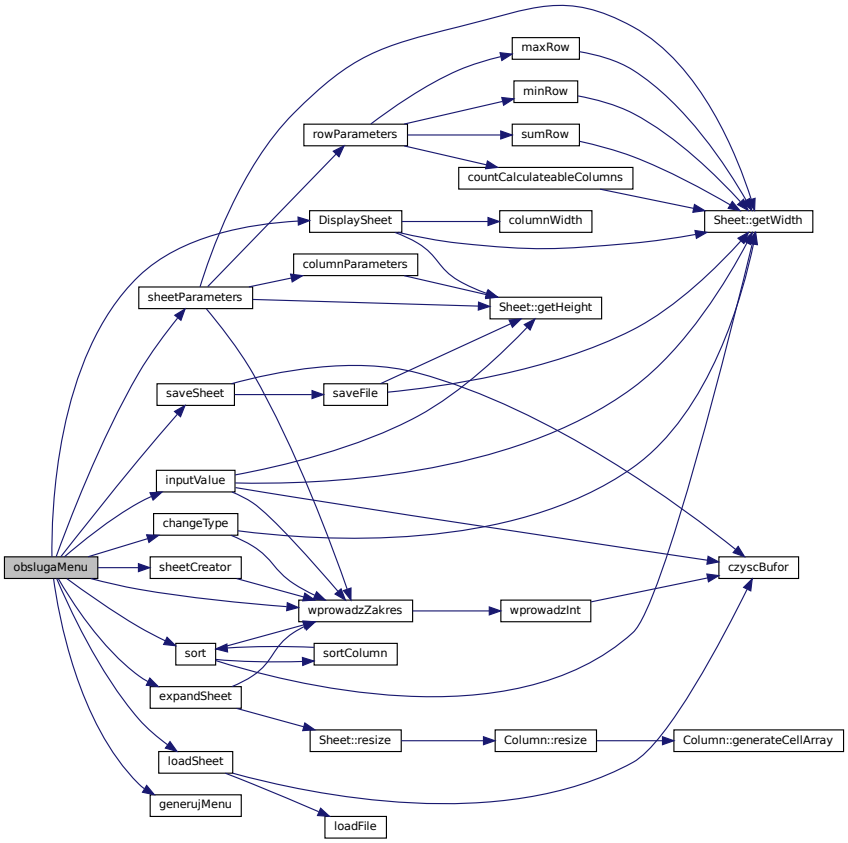
5.22.1.7 obslugaMenu()

```
void obslugaMenu ( )
```

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 )
```

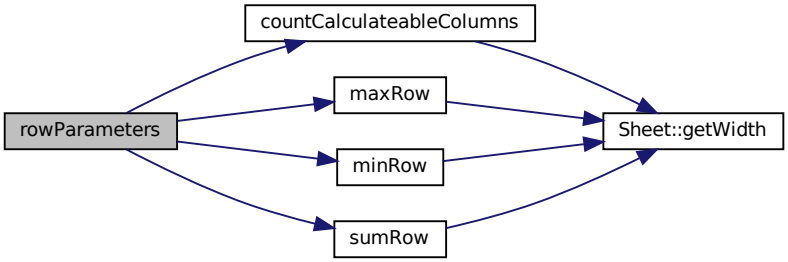
Funkcja od wyznaczania parametrów wiersza arkusza.

Funkcja zwraca w postaci tekstowej wszystkie parametry wybranego wiersza

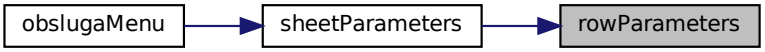
Parametry

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 )
```

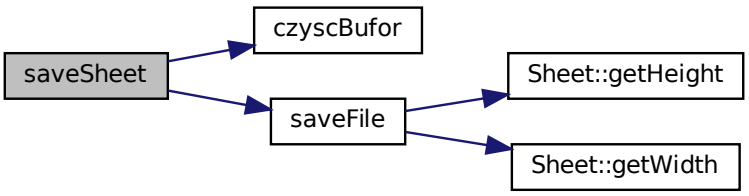
Funkcja menu od zapisu.

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 ( )
```

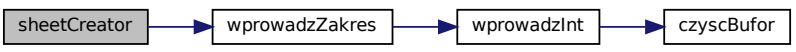
Funkcja tworząca nową tablicę.

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 wywoływań tej funkcji:

5.22.1.11 sheetParameters()

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

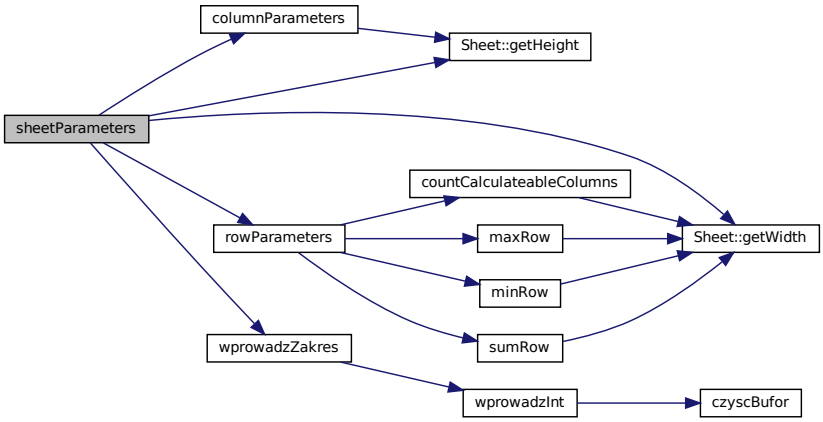
Funkcja menu od wyboru wgzględem czego wyznazcza parametry.

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

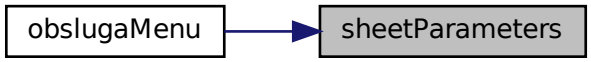
Parametry

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()

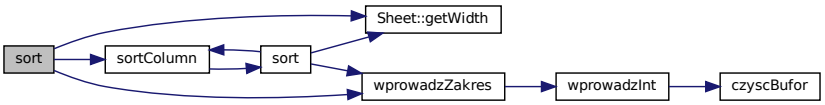
```
void sort (
            Sheet * arkusz )
```

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

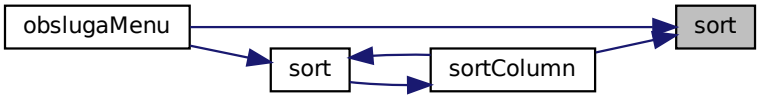
Parametry

in,out	*Arkusz*	którego kolumna będzie sortowana

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

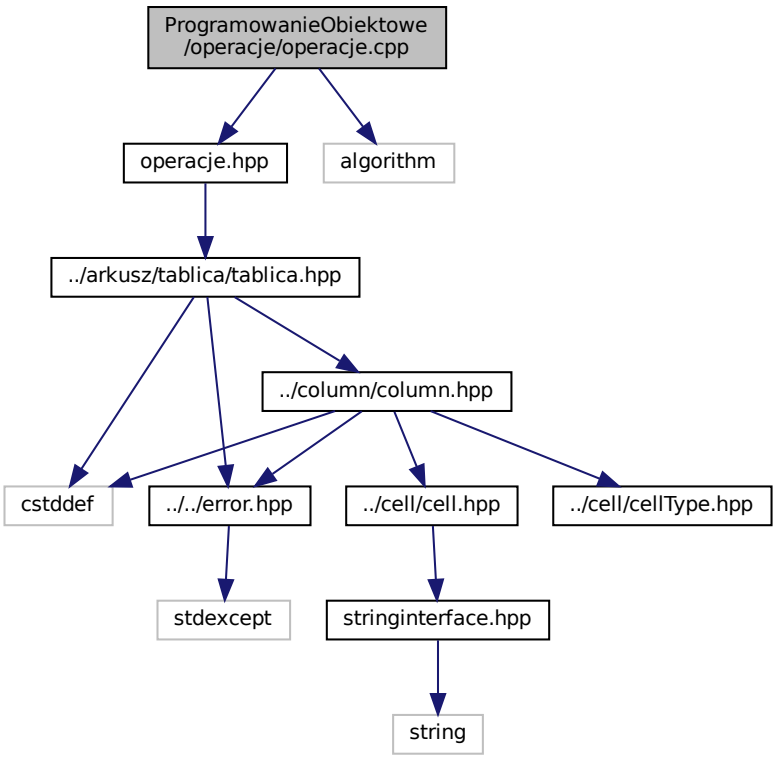


5.23 Dokumentacja pliku ProgramowanieObiektowe/operacje/operacje.cpp

```
#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 kolumne 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

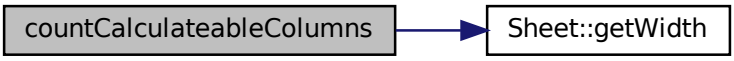
Parametry

sheet	Arkusz którego elementy będą liczone

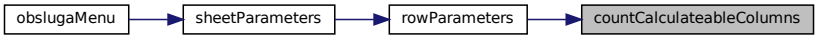
Zwraca

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

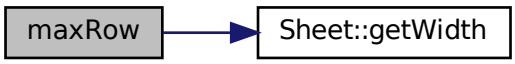
Parametry

in	*sheet*	Arkusz przeznaczony do obliczania parametru
in	*row*	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

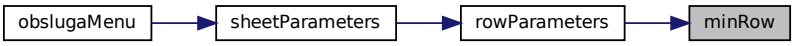
in	*sheet*	Arkusz przeznaczony do obliczania parametru
in	*row*	Wiersz względem którego obliczamy parametr

Zwraca

Zwraca wartość najmniejszą wiersza

Oto graf wywołań dla tej funkcji:

Oto graf wywoływań tej funkcji:



5.23.1.4 sortColumn()

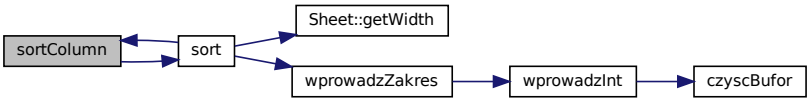
```
void sortColumn (
            Column * column,
            bool descending = false )
```

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

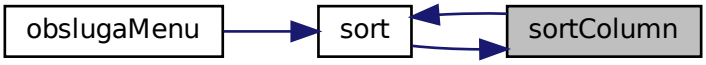
Parametry

in,out	*column*	Kolumna przeznaczona do sortowania
in	*descending*	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

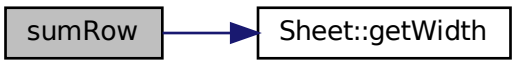
Parametry

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

Zwraca

Zwraca sumę wszystkich elementów wiersza

Oto graf wywołań dla tej funkcji:

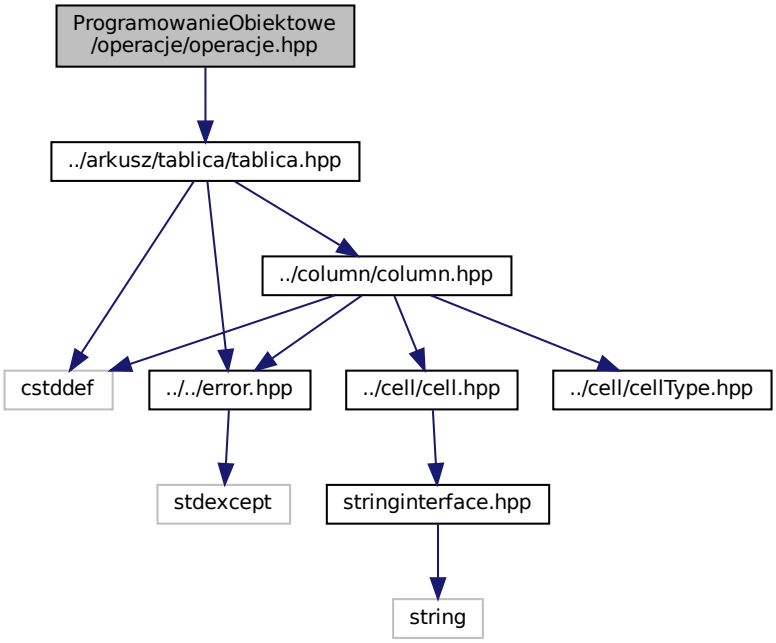


Oto graf wywoływań tej funkcji:

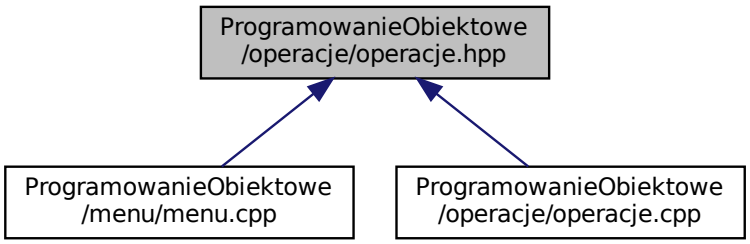
5.24 Dokumentacja pliku ProgramowanieObiektowe/operacje/operacje.hpp

```
#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 kolumne 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

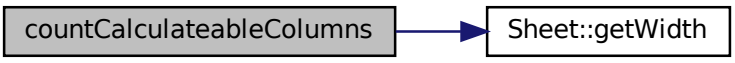
Parametry

sheet	Arkusz którego elementy będą liczone

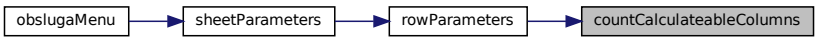
Zwraca

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

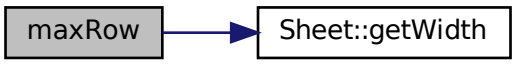
Parametry

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

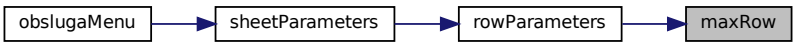
Zwraca

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	*sheet*	Arkusz przeznaczony do obliczania parametru
in	*row*	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()

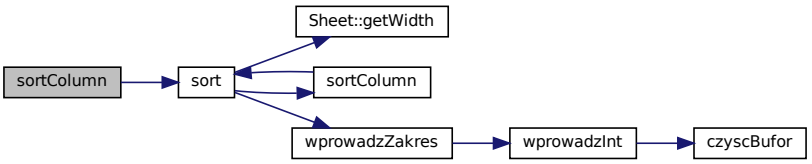
```
void sortColumn (
            Column * column,
            bool descending = false )
```

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

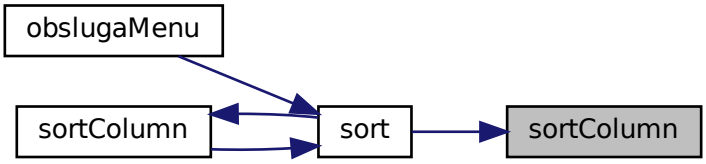
Parametry

in,out	*column*	Kolumna przeznaczona do sortowania
in	*descending*	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

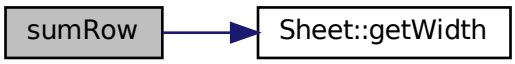
Parametry

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

Zwraca

Zwraca sumę wszystkich elementów wiersza

Oto graf wywołań dla tej funkcji:

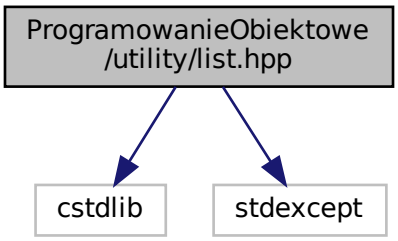


Oto graf wywoływań tej funkcji:

5.25 Dokumentacja pliku ProgramowanieObiektowe/utility/list.hpp

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

Wykres zależności załączania dla list.hpp:



Komponenty

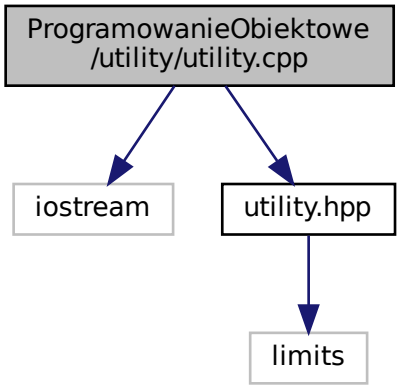
- class List< T >

  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 czyscBufor ()

  Funkcja od czyszczenia buforu strumienia CIN.

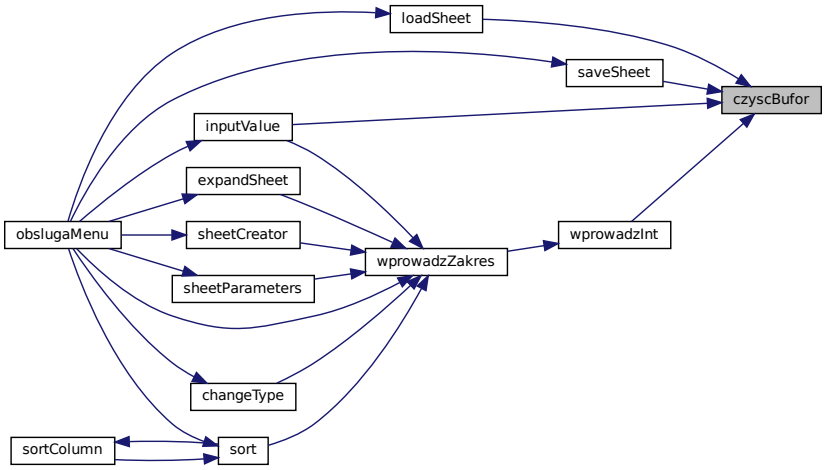
5.26.1 Dokumentacja funkcji

5.26.1.1 czyscBufor()

```
void czyscBufor ( )
```

Funkcja od czyszczenia buforu strumienia CIN.

Funkcja ma za zadanie wyczyścić bufor strumienia wejściowego CIN celem wprowadzenia np. string'a Oto graf wywoływań 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.

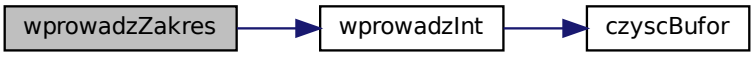
Parametry

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

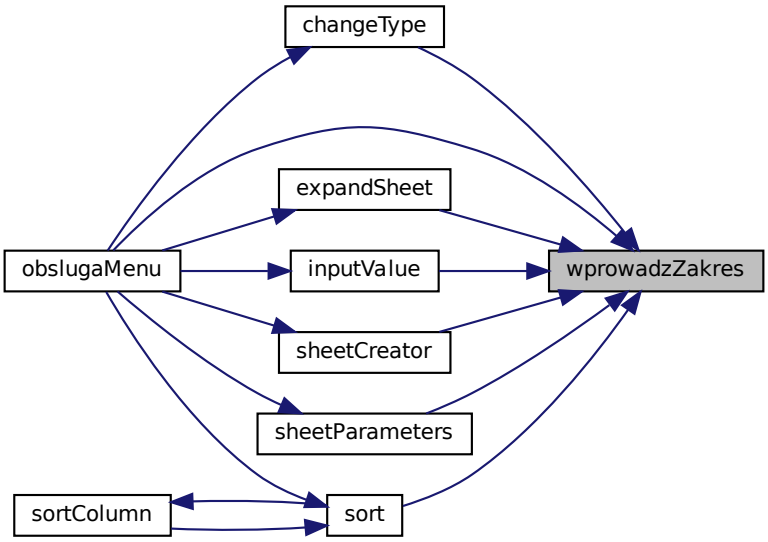
Zwraca

Wartość z zakresu <min; max>

Oto graf wywołań dla tej funkcji:



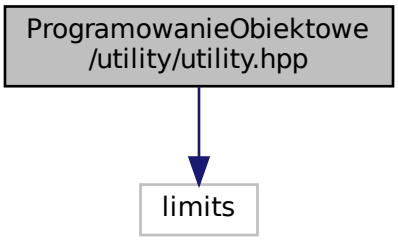
Oto graf wywoływań tej funkcji:



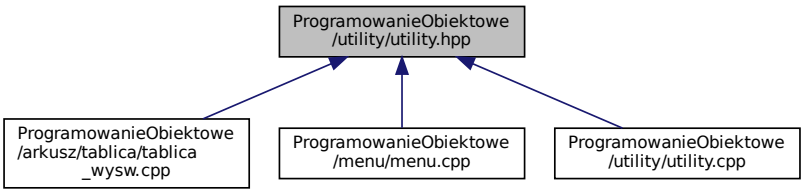
5.27 Dokumentacja pliku ProgramowanieObiektowe/utility/utility.hpp

```
#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 zakresie.
- void czyscBufor ()

  Funkcja od czyszczenia buforu strumienia CIN.

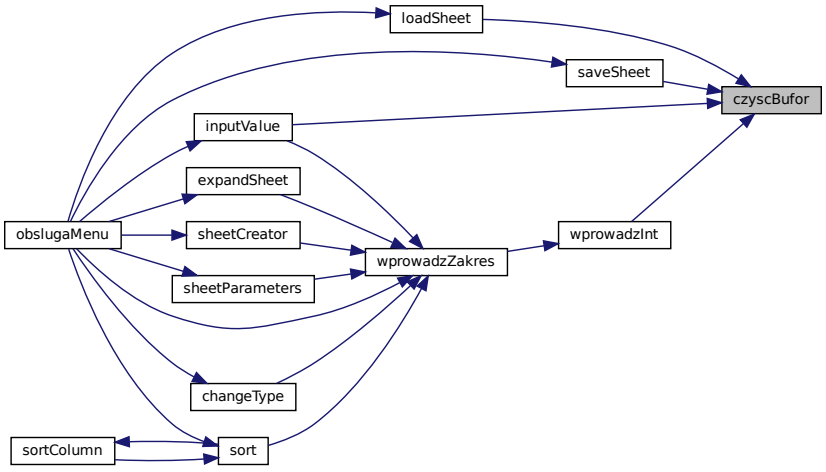
5.27.1 Dokumentacja funkcji

5.27.1.1 czyscBufor()

```
void czyscBufor ( )
```

Funkcja od czyszczenia buforu strumienia CIN.

Funkcja ma za zadanie wyczyścić bufor strumienia wejściowego CIN celem wprowadzenia np. string'a Oto graf wywoływań 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.

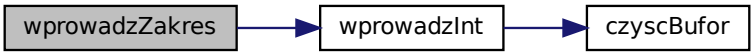
Parametry

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

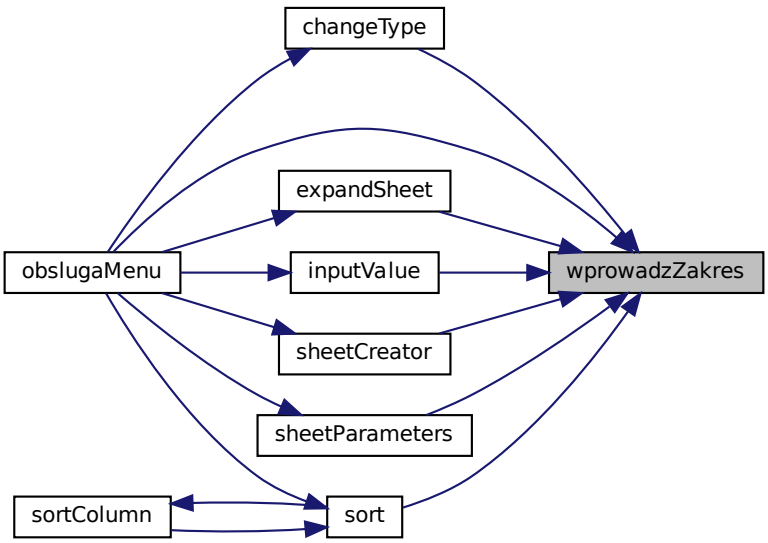
Zwraca

Wartość z zakresu <min; max>

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



Index