

## GRA W ZYCIE

Wygenerowano przez Doxygen 1.8.14



# Spis treści

<b>1</b>	<b>Indeks hierarchiczny</b>	<b>1</b>
1.1	Hierarchia klas . . . . .	1
<b>2</b>	<b>Indeks klas</b>	<b>3</b>
2.1	Lista klas . . . . .	3
<b>3</b>	<b>Indeks plików</b>	<b>5</b>
3.1	Lista plików . . . . .	5
<b>4</b>	<b>Dokumentacja klas</b>	<b>7</b>
4.1	Dokumentacja klasy dane_planszy . . . . .	7
4.1.1	Opis szczegółowy . . . . .	8
4.1.2	Dokumentacja funkcji składowych . . . . .	9
4.1.2.1	get_pocz() . . . . .	9
4.1.2.2	get_roz() . . . . .	10
4.1.2.3	set_pocz() . . . . .	10
4.1.2.4	set_roz() . . . . .	11
4.1.3	Dokumentacja atrybutów składowych . . . . .	12
4.1.3.1	ppoczątkowy . . . . .	12
4.1.3.2	rozmiar . . . . .	12
4.2	Dokumentacja klasy element_listy_sasiadow . . . . .	13
4.2.1	Opis szczegółowy . . . . .	13
4.2.2	Dokumentacja konstruktora i destruktora . . . . .	13
4.2.2.1	element_listy_sasiadow() [1/2] . . . . .	13
4.2.2.2	element_listy_sasiadow() [2/2] . . . . .	14

4.2.3	Dokumentacja funkcji składowych . . . . .	14
4.2.3.1	get_l_sasiadow() . . . . .	14
4.2.3.2	getn() . . . . .	15
4.2.3.3	set_l_sasiadow() . . . . .	15
4.2.3.4	setn() . . . . .	16
4.3	Dokumentacja klasy element_planszy . . . . .	17
4.3.1	Opis szczegółowy . . . . .	18
4.3.2	Dokumentacja funkcji składowych . . . . .	18
4.3.2.1	get_czy_zyje() . . . . .	18
4.3.2.2	getd() . . . . .	19
4.3.2.3	getg() . . . . .	19
4.3.2.4	getl() . . . . .	20
4.3.2.5	getp() . . . . .	21
4.3.2.6	set_czy_zyje() . . . . .	22
4.3.2.7	setd() . . . . .	22
4.3.2.8	setg() . . . . .	23
4.3.2.9	setl() . . . . .	24
4.3.2.10	setp() . . . . .	25
4.4	Dokumentacja klasy komorka . . . . .	26
4.4.1	Opis szczegółowy . . . . .	29
4.4.2	Dokumentacja funkcji składowych . . . . .	29
4.4.2.1	ozyw_zabij_lub_zostaw() . . . . .	29
4.5	Dokumentacja klasy menu . . . . .	30
4.5.1	Opis szczegółowy . . . . .	30
4.5.2	Dokumentacja funkcji składowych . . . . .	30
4.5.2.1	kom_s_czy_s() . . . . .	31
4.5.2.2	kom_s_czy_s_w_petli() . . . . .	31
4.5.2.3	kom_stan_planszy() . . . . .	32
4.5.2.4	kom_start() . . . . .	32
4.5.2.5	kom_wpr_danych() . . . . .	33

4.5.2.6	kom_wpr_danych_w_petli()	33
4.5.2.7	kom_zapis()	34
4.5.2.8	kom_zapis_w_petli()	34
4.5.2.9	petla()	34
4.5.2.10	poprowadz_uzytkownika()	36
4.5.2.11	wybor()	37
4.5.2.12	zatrzymaj_petle()	38
4.6	Dokumentacja klasy plansza	39
4.6.1	Opis szczegółowy	41
4.6.2	Dokumentacja konstruktora i destruktora	41
4.6.2.1	~plansza()	41
4.6.3	Dokumentacja funkcji składowych	41
4.6.3.1	operator int()	42
4.6.3.2	rozszerz_plansze_w_dol()	42
4.6.3.3	rozszerz_plansze_w_gore()	43
4.6.3.4	rozszerz_plansze_w_lewo()	44
4.6.3.5	rozszerz_plansze_w_prawo()	45
4.6.3.6	stworz_element()	46
4.6.3.7	stworz_podstawowa_plansze()	47
4.6.3.8	usun_plansze()	48
4.7	Dokumentacja klasy sasiedzi	49
4.7.1	Opis szczegółowy	51
4.7.2	Dokumentacja konstruktora i destruktora	51
4.7.2.1	sasiedzi()	51
4.7.2.2	~sasiedzi()	51
4.7.3	Dokumentacja funkcji składowych	51
4.7.3.1	dodaj_na_koniec_listy_sasiadow()	51
4.7.3.2	operator+=()	52
4.7.3.3	policz_sasiadow()	53
4.7.3.4	przesun_wezykiem_w_dol()	54

4.7.3.5	rozszerz_w_dobra_strone()	55
4.7.3.6	sprawdz_czy_krawedz()	56
4.7.3.7	usun_liste_sasiadow()	57
4.7.4	Dokumentacja atrybutów składowych	58
4.7.4.1	ob	58
4.7.4.2	pglowa	58
4.8	Dokumentacja klasy wspolrzedne	59
4.8.1	Opis szczegółowy	61
4.8.2	Dokumentacja funkcji składowych	61
4.8.2.1	podaj_rozmiar_planszy()	61
4.8.2.2	wypelnij_tab_wsp()	62
4.8.2.3	wypelnij_tab_wsp_z_pliku()	62
4.9	Dokumentacja klasy wypelnij	63
4.9.1	Opis szczegółowy	66
4.9.2	Dokumentacja konstruktora i destruktora	66
4.9.2.1	wypelnij()	66
4.9.2.2	~wypelnij()	66
4.9.3	Dokumentacja funkcji składowych	66
4.9.3.1	get_el_punktow()	66
4.9.3.2	get_punkty()	67
4.9.3.3	operator[]()	68
4.9.3.4	set_el_punktow()	68
4.9.3.5	set_punkty()	69
4.9.3.6	stworz_tab_wsp()	70
4.9.3.7	usun_tab_wsp()	70
4.9.3.8	wypelnij_plansze()	71
4.9.4	Dokumentacja atrybutów składowych	72
4.9.4.1	punkty	72
4.10	Dokumentacja klasy wyswietl	72
4.10.1	Opis szczegółowy	73
4.10.2	Dokumentacja funkcji składowych	74
4.10.2.1	wyswietl_plansze()	74
4.10.2.2	zapisz_plansze()	74

<b>5 Dokumentacja plików</b>	<b>77</b>
5.1 Dokumentacja pliku klasa_dane_planszy.h . . . . .	77
5.2 Dokumentacja pliku klasa_element_listy_sasiadow.h . . . . .	77
5.3 Dokumentacja pliku klasa_element_planszy.h . . . . .	78
5.4 Dokumentacja pliku klasa_komorka.h . . . . .	78
5.5 Dokumentacja pliku klasa_menu.h . . . . .	78
5.6 Dokumentacja pliku klasa_plansza.h . . . . .	79
5.7 Dokumentacja pliku klasa_sasiedzi.h . . . . .	79
5.8 Dokumentacja pliku klasa_wspolrzedne.h . . . . .	80
5.9 Dokumentacja pliku klasa_wypelnij.h . . . . .	80
5.10 Dokumentacja pliku klasa_wyswietl.h . . . . .	80
5.11 Dokumentacja pliku komunikaty_wyjatki.cpp . . . . .	81
5.11.1 Dokumentacja funkcji . . . . .	81
5.11.1.1 kom_brak_dost_do_ob_dane_planszy() . . . . .	82
5.11.1.2 kom_brak_ob_element_listy_sasiadow() . . . . .	82
5.11.1.3 kom_brak_ob_element_planszy() . . . . .	82
5.11.1.4 kom_brak_ob_wypelnij() . . . . .	83
5.11.1.5 kom_brak_ob_wyswietl() . . . . .	83
5.11.1.6 kom_brak_pliku() . . . . .	84
5.11.1.7 kom_brak_wsp_y() . . . . .	84
5.11.1.8 kom_wsp_spoza_planszy() . . . . .	85
5.11.1.9 wyswietl_komunikat() . . . . .	85
5.12 Dokumentacja pliku komunikaty_wyjatki.h . . . . .	85
5.12.1 Dokumentacja funkcji . . . . .	86
5.12.1.1 kom_brak_dost_do_ob_dane_planszy() . . . . .	86
5.12.1.2 kom_brak_ob_element_listy_sasiadow() . . . . .	86
5.12.1.3 kom_brak_ob_element_planszy() . . . . .	87
5.12.1.4 kom_brak_ob_wypelnij() . . . . .	87
5.12.1.5 kom_brak_ob_wyswietl() . . . . .	88
5.12.1.6 kom_brak_pliku() . . . . .	88

5.12.1.7	kom_brak_wsp_y()	88
5.12.1.8	kom_wsp_spoza_planszy()	89
5.12.1.9	wyswietl_komunikat()	89
5.13	Dokumentacja pliku main.cpp	90
5.13.1	Dokumentacja funkcji	90
5.13.1.1	main()	90
5.14	Dokumentacja pliku metody_dane_planszy.cpp	91
5.15	Dokumentacja pliku metody_element_listy_sasiadow.cpp	91
5.16	Dokumentacja pliku metody_element_planszy.cpp	92
5.17	Dokumentacja pliku metody_komorka.cpp	92
5.18	Dokumentacja pliku metody_menu.cpp	93
5.19	Dokumentacja pliku metody_plansza.cpp	93
5.20	Dokumentacja pliku metody_sasiedzi.cpp	94
5.21	Dokumentacja pliku metody_wspolrzedne.cpp	94
5.22	Dokumentacja pliku metody_wypelnij.cpp	94
5.23	Dokumentacja pliku metody_wyswietl.cpp	95
5.24	Dokumentacja pliku operatory.cpp	95
5.24.1	Dokumentacja funkcji	95
5.24.1.1	operator<<() [1/3]	96
5.24.1.2	operator<<() [2/3]	96
5.24.1.3	operator<<() [3/3]	97
5.24.1.4	operator>>()	98
5.25	Dokumentacja pliku operatory.h	99
5.25.1	Dokumentacja funkcji	99
5.25.1.1	operator<<() [1/3]	100
5.25.1.2	operator<<() [2/3]	100
5.25.1.3	operator<<() [3/3]	101
5.25.1.4	operator>>()	102



# Rozdział 1

## Indeks hierarchiczny

### 1.1 Hierarchia klas

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

dane_planszy . . . . .	7
plansza . . . . .	39
wypelnij . . . . .	63
wspolrzedne . . . . .	59
wyswietl . . . . .	72
element_listy_sasiadow . . . . .	13
element_planszy . . . . .	17
menue . . . . .	30
sasiedzi . . . . .	49
komorka . . . . .	26



## Rozdział 2

# Indeks klas

### 2.1 Lista klas

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

dane_planszy . . . . .	7
element_listy_sasiadow . . . . .	13
element_planszy . . . . .	17
komorka . . . . .	26
menue . . . . .	30
plansza . . . . .	39
sasiedzi . . . . .	49
wspolrzedne . . . . .	59
wypelnij . . . . .	63
wyswietl . . . . .	72



## Rozdział 3

# Indeks plików

### 3.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

<a href="#">klasa_dane_planszy.h</a>	77
<a href="#">klasa_element_listy_sasiadow.h</a>	77
<a href="#">klasa_element_planszy.h</a>	78
<a href="#">klasa_komorka.h</a>	78
<a href="#">klasa_menu.h</a>	78
<a href="#">klasa_plansza.h</a>	79
<a href="#">klasa_sasiedzi.h</a>	79
<a href="#">klasa_wspolrzedne.h</a>	80
<a href="#">klasa_wypelnij.h</a>	80
<a href="#">klasa_wyswietl.h</a>	80
<a href="#">komunkiaty_wyjatki.cpp</a>	81
<a href="#">komunkiaty_wyjatki.h</a>	85
<a href="#">main.cpp</a>	90
<a href="#">metody_dane_planszy.cpp</a>	91
<a href="#">metody_element_listy_sasiadow.cpp</a>	91
<a href="#">metody_element_planszy.cpp</a>	92
<a href="#">metody_komorka.cpp</a>	92
<a href="#">metody_menu.cpp</a>	93
<a href="#">metody_plansza.cpp</a>	93
<a href="#">metody_sasiedzi.cpp</a>	94
<a href="#">metody_wspolrzedne.cpp</a>	94
<a href="#">metody_wypelnij.cpp</a>	94
<a href="#">metody_wyswietl.cpp</a>	95
<a href="#">operatory.cpp</a>	95
<a href="#">operatory.h</a>	99



## Rozdział 4

# Dokumentacja klas

### 4.1 Dokumentacja klasy dane\_planszy

```
#include <klasa_dane_planszy.h>
```

Diagram dziedziczenia dla dane\_planszy

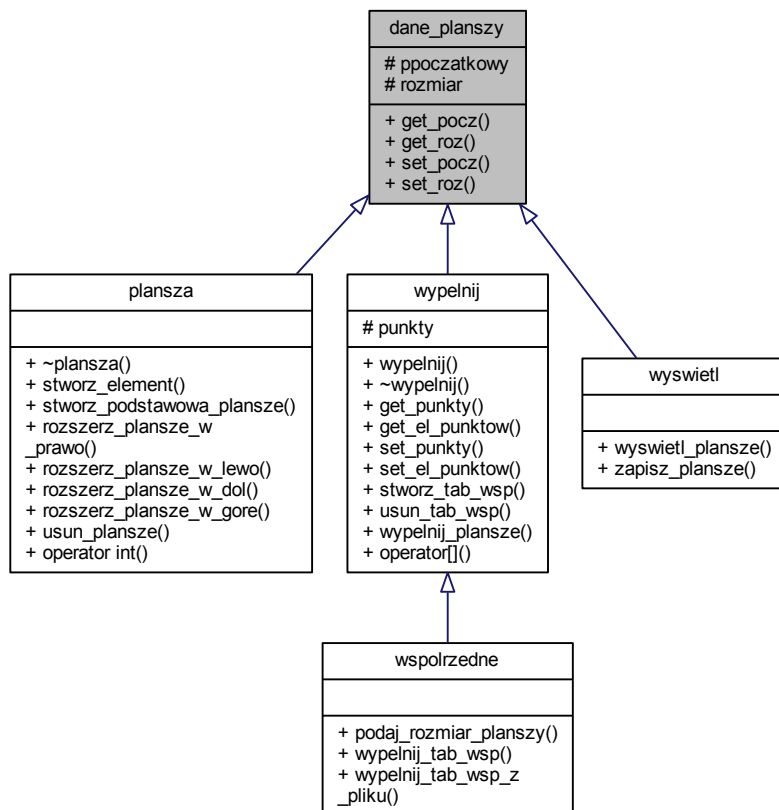
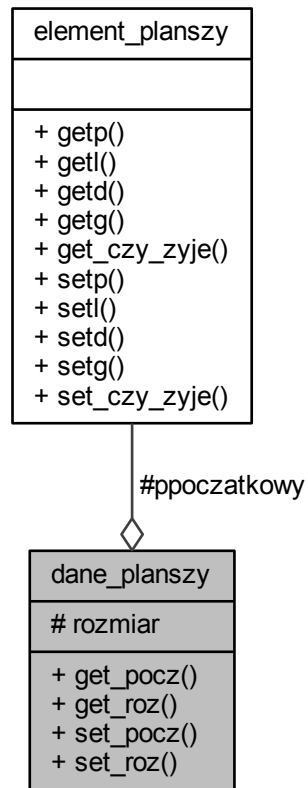


Diagram współpracy dla dane\_planszy:



### Metody publiczne

- [element\\_planszy](#) \* [get\\_pocz](#) ()
- int [get\\_roz](#) ()
- void [set\\_pocz](#) ([element\\_planszy](#) \*pp)
- void [set\\_roz](#) (int r)

### Atrybuty chronione

- [element\\_planszy](#) \* [ppoczatkowy](#)
- int [rozmiar](#)

#### 4.1.1 Opis szczegółowy

Klasa zawiera podstawowe dane planszy do gry w życie. Jest klasa bazowa klas: [plansza](#), [wyjwiatl](#), [wypelnij](#)(po której dziedziczy klasa [wspolrzedne](#)).



## 4.1.2 Dokumentacja funkcji składowych

### 4.1.2.1 get\_pocz()

```
element_planszy * dane_planszy::get_pocz ( )
```

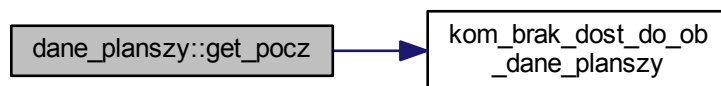
Metoda zwraca pole ppoczątkowy.

Metoda została stworzona aby uzyskać dostęp do pola ppoczątkowy przez klasy niezaprzyjżaznione i niepochoďne. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

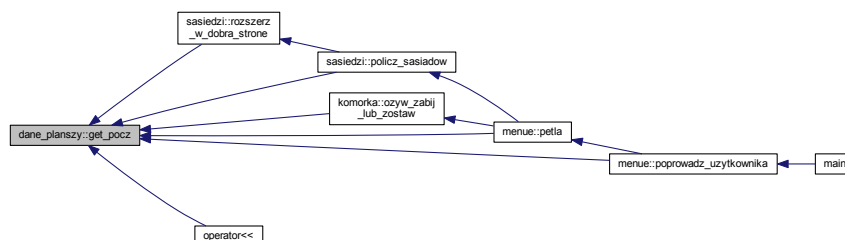
#### Zwraca

ppoczątkowy, jeśli jest dostęp do obiektu klasy `dane_planszy`. Jeśli nie, zwraca nullptr.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.1.2.2 get\_roz()

```
int dane_planszy::get_roz ( )
```

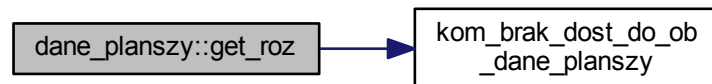
Metoda zwraca pole rozmiar.

Metoda została stworzona aby uzyskać dostęp do pola rozmiar przez klasy niezaprzyjznione i nie pochodne. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

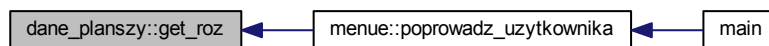
##### Zwraca

rozmiar, jeśli jest dostęp do obiektu klasy `dane_planszy`. Jeśli nie, zwraca -1.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.1.2.3 set\_pocz()

```
void dane_planszy::set_pocz (
    element_planszy * pp )
```

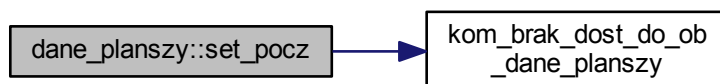
Metoda ustawia pole ppoczątkowy.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę wartości pola `ppoczątkowy` przez klasy niezaprzyjznione i nie pochodne. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

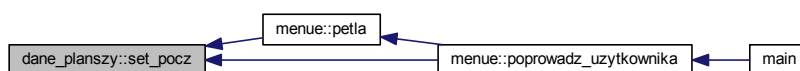
## Parametry

<i>pp</i>	wskaznik na komorke w lewym gornym rogu.
-----------	--

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



## 4.1.2.4 set\_roz()

```
void dane_planszy::set_roz (
    int r )
```

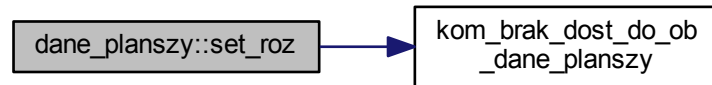
Metoda ustawia pole rozmiar.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę wartości pola rozmiar przez klasy niezaprzyjawnione i nie pochodne. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

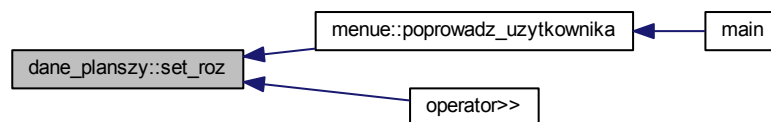
## Parametry

<i>r</i>	długość boku podstawowej planszy kwadratowej.
----------	---

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



### 4.1.3 Dokumentacja atrybutów składowych

#### 4.1.3.1 ppoczątkowy

```
element_planszy* dane_planszy::ppoczątkowy [protected]
```

Pole `ppoczątkowy` jest wskaźnikiem na pierwszy element planszy, który znajduje się w lewym górnym rogu.

#### 4.1.3.2 rozmiar

```
int dane_planszy::rozmiar [protected]
```

Pole `rozmiar` przechowuje rozmiar kwadratowej początkowej planszy (długość boku), którą podał użytkownik, tak aby zmieściły się w niej wszystkie komórki do ożywienia.

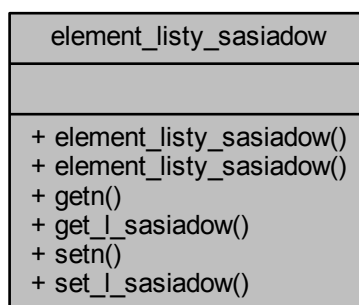
Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_dane\\_planszy.h](#)
- [metody\\_dane\\_planszy.cpp](#)

## 4.2 Dokumentacja klasy `element_listy_sasiadow`

```
#include <klasa_element_listy_sasiadow.h>
```

Diagram współpracy dla `element_listy_sasiadow`:



### Metody publiczne

- `element_listy_sasiadow ()`
- `element_listy_sasiadow (element_listy_sasiadow *pn, int l_s)`
- `element_listy_sasiadow * getn ()`
- `int get_l_sasiadow ()`
- `void setn (element_listy_sasiadow *pn)`
- `void set_l_sasiadow (int l_s)`

### 4.2.1 Opis szczegółowy

Klasa reprezentuje element listy sasiadow. Obiekt klasy zawiera dwa pola : wskaźnik na następny element listy oraz liczbę sasiadow danego elementu planszy.

### 4.2.2 Dokumentacja konstruktora i destruktora

#### 4.2.2.1 `element_listy_sasiadow()` [1/2]

```
element_listy_sasiadow::element_listy_sasiadow ( )
```

Konstruktor klasy `element_listy_sasiadow`.

Nie ustawia żadnych wartości. Został stworzony na potrzeby konstruktora wieloargumentowego.

#### 4.2.2.2 `element_listy_sasiadow()` [2/2]

```
element_listy_sasiadow::element_listy_sasiadow (
    element_listy_sasiadow * pn,
    int l_s )
```

Konstruktor klasy `element_listy_sasiadow`.

Konstruktor przypisuje polom wartosci podane jako argumenty. Zostal stworzony, aby szybciej tworzyc obiekty klasy `element_listy_sasiadow`.

### 4.2.3 Dokumentacja funkcji składowych

#### 4.2.3.1 `get_l_sasiadow()`

```
int element_listy_sasiadow::get_l_sasiadow ( )
```

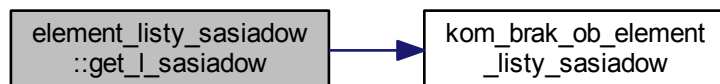
Metoda zwraca pole `liczba_sasiadow`.

Metoda zostala stworzona aby uzyskac dostep do pola `liczba_sasiadow` przez klasy niezaprzyjznione i nie pochodne. Jesli cos poszlo nie tak i nie ma dostepu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjatek.

#### Zwraca

rozmiar, jesli jest dotep do obiektu klasy `dane_planszy`. Jesli nie, zwraca -1.

Oto graf wywołań dla tej funkcji:



4.2.3.2 `getn()`

```
element_listy_sasiadow * element_listy_sasiadow::getn ( )
```

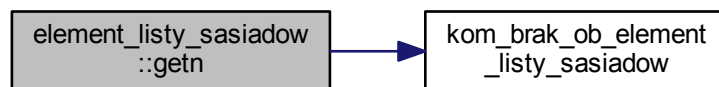
Metoda zwraca pole `pnext`.

Metoda została stworzona aby uzyskać dostęp do pola `pnext` przez klasy niezaprzyjznione i niepo pochodne. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

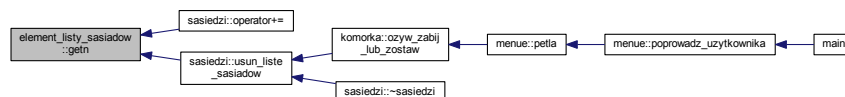
## Zwraca

ppoczątkowy, jeśli jest dostęp do obiektu klasy `dane_planszy`. Jeśli nie, zwraca `nullptr`.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

4.2.3.3 `set_l_sasiadow()`

```
void element_listy_sasiadow::set_l_sasiadow (
    int l_s )
```

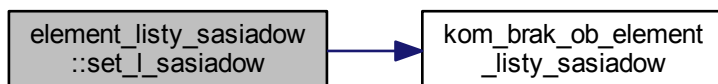
Metoda ustawia pole `liczba_sasiadow`.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę pola `liczba_sasiadow` przez klasy niezaprzyjznione i niepo pochodne. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

## Parametry

<i>l</i> ↔	dlugosc boku podstawowej planszy kwadratowej.
<i>_</i> ↔	
<i>s</i>	

Oto graf wywołań dla tej funkcji:



## 4.2.3.4 setn()

```
void element_listy_sasiadow::setn (
    element_listy_sasiadow * pn )
```

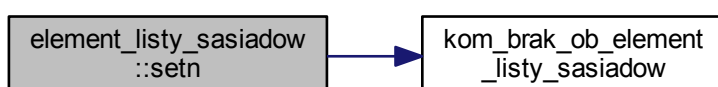
Metoda ustawia pole pnext.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę wartości pola pnext przez klasy niezaprzyjawnione i niepo pochodne. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy [dane\\_planszy](#) (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

## Parametry

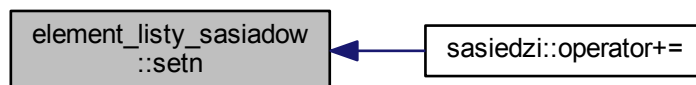
<i>pn</i>	wskaznik na następny element listy sasiadow.
-----------	--

Oto graf wywołań dla tej funkcji:





Oto graf wywoływań tej funkcji:



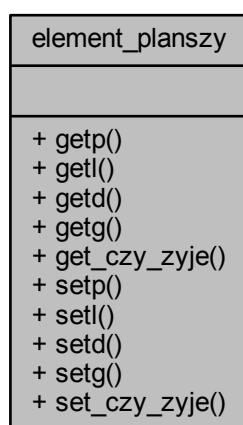
Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_element\\_listy\\_sasiadow.h](#)
- [metody\\_element\\_listy\\_sasiadow.cpp](#)

### 4.3 Dokumentacja klasy element\_planszy

```
#include <klasa_element_planszy.h>
```

Diagram współpracy dla element\_planszy:



#### Metody publiczne

- [element\\_planszy \\* getp \(\)](#)
- [element\\_planszy \\* getl \(\)](#)
- [element\\_planszy \\* getd \(\)](#)
- [element\\_planszy \\* getg \(\)](#)
- [bool get\\_czy\\_zyje \(\)](#)
- [void setp \(element\\_planszy \\*pp\)](#)
- [void setl \(element\\_planszy \\*pl\)](#)
- [void setd \(element\\_planszy \\*pd\)](#)
- [void setg \(element\\_planszy \\*pg\)](#)
- [void set\\_czy\\_zyje \(bool zyje\)](#)

### 4.3.1 Opis szczegółowy

Klasa reprezentuje element planszy do gry w życie. Obiekt klasy zawiera trzy pola wskaźnikowe zawierające adresy sąsiadów danego elementu oraz pole typu bool, które informuje o tym, czy komórka żyje.

### 4.3.2 Dokumentacja funkcji składowych

#### 4.3.2.1 `get_czy_zyje()`

```
bool element_planszy::get_czy_zyje ( )
```

Metoda zwraca pole `czy_zyje`.

Metoda została stworzona aby uzyskać dostęp do pola `pgorny` przez klasy niezaprzyjżaznione i niepo pochodne. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

#### Zwraca

`pgorny`, jeśli jest dostęp do obiektu klasy `dane_planszy`. Jeśli nie, zwraca `nullptr`.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



## 4.3.2.2 getd()

```
element_planszy * element_planszy::getd ( )
```

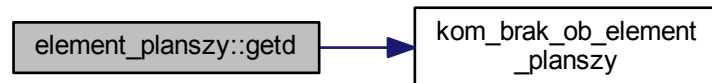
Metoda zwraca pole pdolny.

Metoda została stworzona aby uzyskać dostęp do pola pdolny przez klasy niezaprzyjżaznione i nie pochodne. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

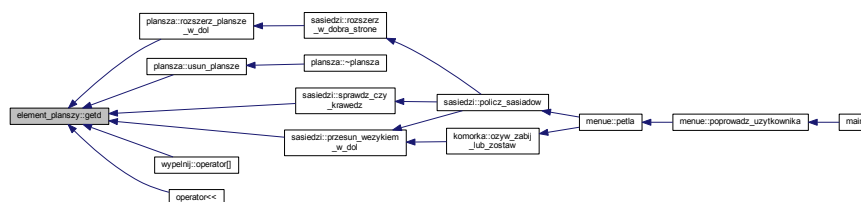
Zwraca

pdolny, jeśli jest dostęp do obiektu klasy `dane_planszy`. Jeśli nie, zwraca nullptr.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



## 4.3.2.3 getg()

```
element_planszy * element_planszy::getg ( )
```

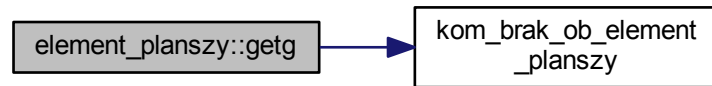
Metoda zwraca pole pgorny.

Metoda została stworzona aby uzyskać dostęp do pola pgorny przez klasy niezaprzyjżaznione i nie pochodne. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

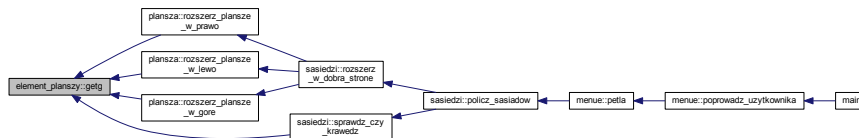
**Zwraca**

pgorny, jesli jest dotep do obiektu klasy `dane_planszy`. Jesli nie, zwraca nullptr.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

**4.3.2.4 getl()**

```
element_planszy * element_planszy::getl ( )
```

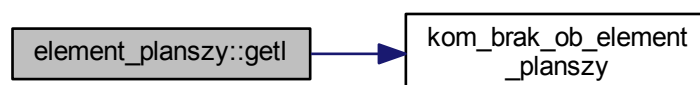
Metoda zwraca pole plewy.

Metoda została stworzona aby uzyskać dostęp do pola plewy przez klasy niezaprzyjżnione i niepo pochodne. Jesli cos poszlo nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

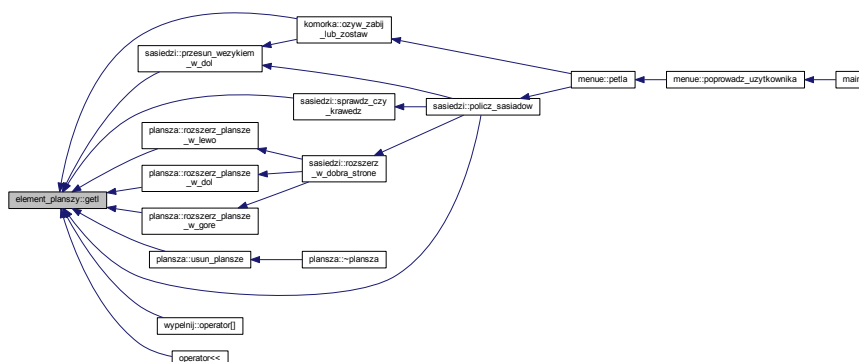
**Zwraca**

plewy, jesli jest dotep do obiektu klasy `dane_planszy`. Jesli nie, zwraca nullptr.

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



#### 4.3.2.5 getp()

```
element_planszy * element_planszy::getp ( )
```

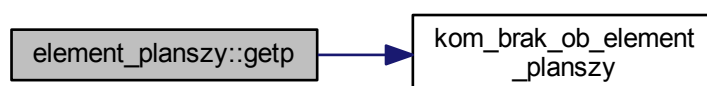
Metoda zwraca pole pprawy.

Metoda została stworzona aby uzyskać dostęp do pola pprawy przez klasy niezaprzyjżaznione i nie pochodne. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

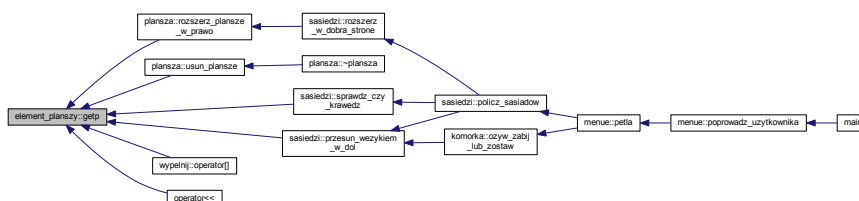
Zwraca

ppraw, jeśli jest dostęp do obiektu klasy `dane_planszy`. Jeśli nie, zwraca nullptr.

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



#### 4.3.2.6 set\_czy\_zyje()

```
void element_planszy::set_czy_zyje (
    bool zyje )
```

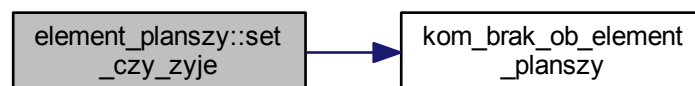
Metoda ustawia pole czy\_zyje.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę pola czy\_zyje przez klasy niezaprzyjżaznione i nie pochodne. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy [dane\\_planszy](#) (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

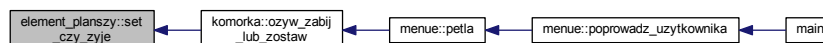
##### Parametry

<a href="#">zyje</a>	stan w jaki komórka ma być wprowadzona (ożywiona lub zabita).
----------------------	---

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



#### 4.3.2.7 setd()

```
void element_planszy::setd (
    element_planszy * pd )
```

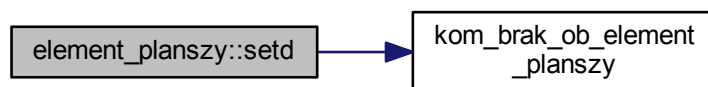
Metoda ustawia pole pdolny.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę pola pdolny przez klasy niezaprzyjżaznione i nie pochodne. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy [dane\\_planszy](#) (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

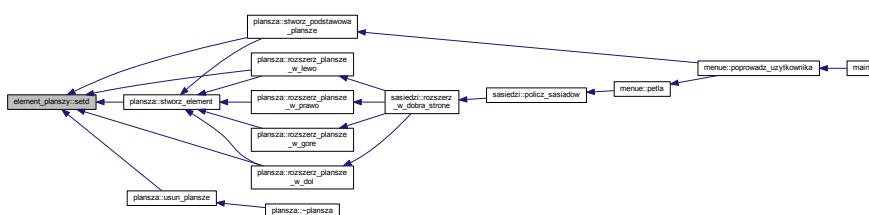
##### Parametry

<i>pd</i>	wskaznik na element na dole.
-----------	------------------------------

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.3.2.8 setg()

```
void element_planszy::setg (
    element_planszy * pg )
```

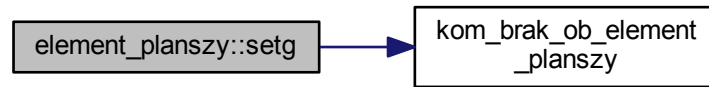
Metoda ustawia pole pgorny.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę pola pgorny przez klasy niezaprzyjane i niepochothane. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

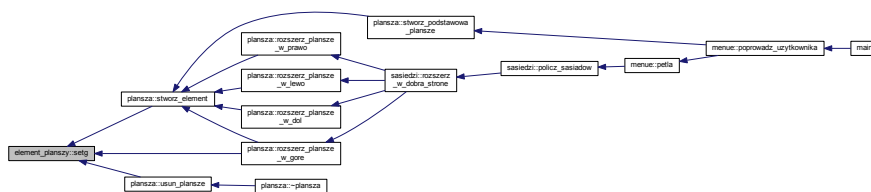
#### Parametry

<code>pg</code>	wskaznik na element na gorze.
-----------------	-------------------------------

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.3.2.9 setl()

```
void element_planszy::setl (
    element_planszy * pl )
```

Metoda ustawia pole plewy.

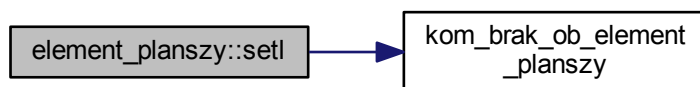
Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę pola plewy przez klasy niezaprzyjżnione i niepo pochodne. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

#### Parametry

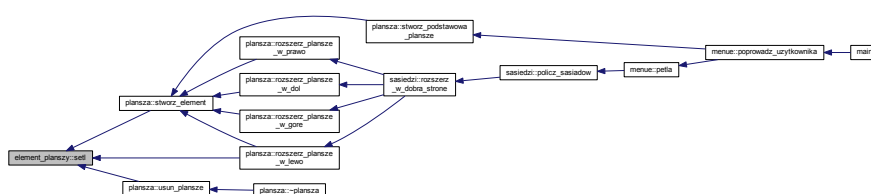
<i>pl</i>	wskaznik na element po lewej.
-----------	-------------------------------



Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.3.2.10 setp()

```
void element_planszy::setp (
    element_planszy * pp )
```

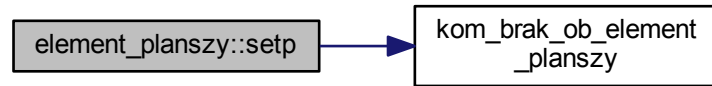
Metoda ustawia pole pprawy.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę pola poprzez klasy niezaprzyjane i nie pochodne. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `dane_planszy` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

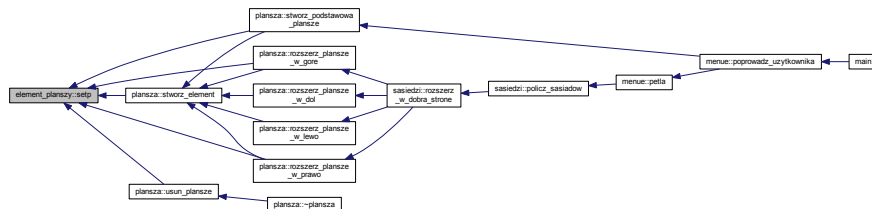
### Parametry

<i>pp</i>	wskaznik na element po prawej.
-----------	--------------------------------

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_element\\_planszy.h](#)
- [metody\\_element\\_planszy.cpp](#)

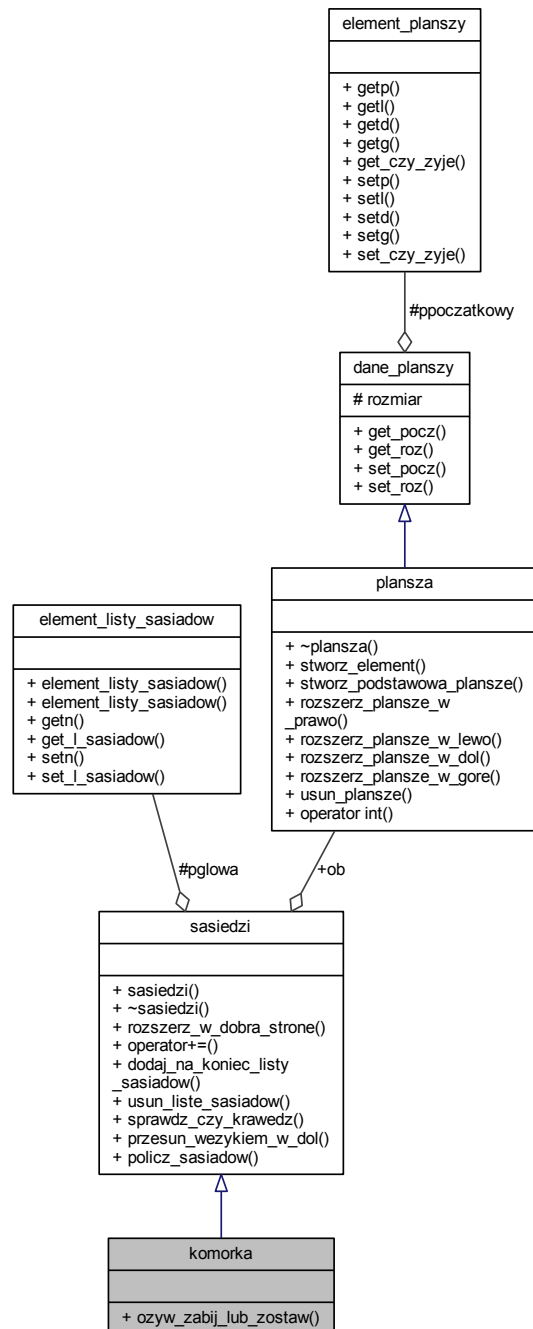
## 4.4 Dokumentacja klasy komorka

```
#include <klasa_komorka.h>
```

Diagram dziedziczenia dla komorka



Diagram współpracy dla komórka:



## Metody publiczne

- void [ozyw\\_zabij\\_lub\\_zostaw](#) ()

## Dodatkowe Dziedziczone Składowe

### 4.4.1 Opis szczegółowy

Klasa sluzy do "ozywania i zabijania komorek". Dziedziczy po klasie sasiedzi i dodaje tylko jedna metode. Nie zawiera dodatkowych pol.

### 4.4.2 Dokumentacja funkcji składowych

#### 4.4.2.1 ozyw\_zabij\_lub\_zostaw()

```
void komorka::ozyw_zabij_lub_zostaw ( )
```

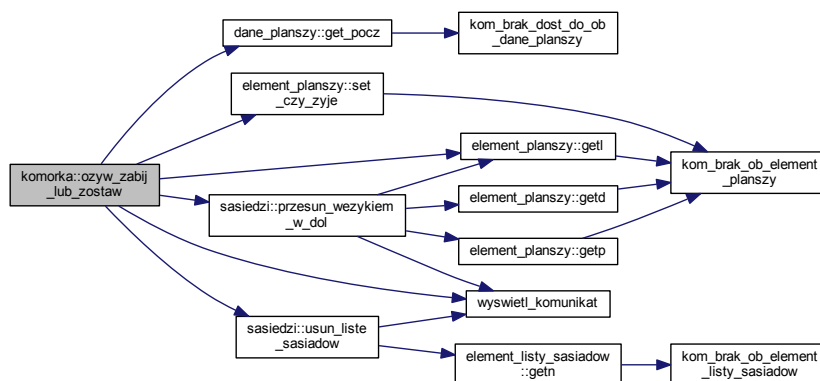
Metoda w zaleznosci od ilosci sasiadow komorki zabija ozywia lub zostawia w poprzednim stanie komorke.

Metoda uzywa dwoch wskaznikow pomocniczych. Pierwszy, tmp\_lista jest uzyty do przechodzenia listy sasiadow i odczytywania z niej wartosci. Drugi, tmp jest uzyty do przechodzenia planszy "wezykiem w dol"(wywolana jest do tego metoda klasy sasiedzi przesun\_wezykiem\_w\_dol) i sprawdzania oraz ustawiania stanu komorek w zaleznosci od ilosci sasiadow pod wskaznikiem tmp\_lista. Metoda lapie ewentualne wyjatki wyrzucone przez uzyte w niej inne metody.

Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_komorka.h](#)
- [metody\\_komorka.cpp](#)

## 4.5 Dokumentacja klasy menu

```
#include <klasa_menu.h>
```

Diagram współpracy dla menu:



### Statyczne metody publiczne

- static void `kom_wpr_danych ()`
- static void `kom_wpr_danych_w_petli ()`
- static void `kom_s_czy_s ()`
- static void `kom_s_czy_s_w_petli ()`
- static void `kom_zapis ()`
- static void `kom_zapis_w_petli ()`
- static void `kom_stan_planszy ()`
- static void `kom_start ()`
- static int `wybor (int &zmiana_wyb)`
- static void `petla (komorka ko, wyswietl wys)`
- static void `zatrzymaj_petle ()`
- static void `poprowadz_uzytkownika ()`

#### 4.5.1 Opis szczegółowy

Klasa służy do łączenia wszystkich operacji, aby gra w życie działała poprawnie oraz do komunikacji z użytkownikiem.

#### 4.5.2 Dokumentacja funkcji składowych

## 4.5.2.1 kom\_s\_czy\_s()

```
void menue::kom_s_czy_s ( ) [static]
```

Metoda wyświetla komentarz na strumień wyjściowy.

Wyświetlany komentarz prosi użytkownika o wybór, czy użytkownik chce rozpocząć grę w życie z ożywionymi przez siebie komórkami, czy wyłączyć program i coś jeszcze zmienić.

**Zwraca**

nic

Oto graf wywołań tej funkcji:



## 4.5.2.2 kom\_s\_czy\_s\_w\_petli()

```
void menue::kom_s_czy_s_w_petli ( ) [static]
```

Metoda wyświetla komentarz na strumień wyjściowy.

Wyświetlany komentarz dotyczy wyboru, czy użytkownik chce wystartować grę, czy zakończyć program i coś zmienić. Jest on wyświetlany w pętli, która kończy się gdy użytkownik poda poprawną odpowiedź (1 lub 2).

**Zwraca**

nic

Oto graf wywołań tej funkcji:



#### 4.5.2.3 kom\_stan\_planszy()

```
void menue::kom_stan_planszy ( ) [static]
```

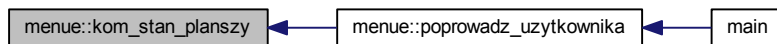
Metoda wyświetla komentarz na strumień wyjściowy.

Wyswietlany komentarz informuje, że zaraz zostanie wyświetlony aktualny stan planszy. Metoda jest użyta przed wyświetleniem planszy w wersji zaraz po wprowadzeniu danych przez użytkownika.

**Zwraca**

nic

Oto graf wywołań tej funkcji:



#### 4.5.2.4 kom\_start()

```
void menue::kom_start ( ) [static]
```

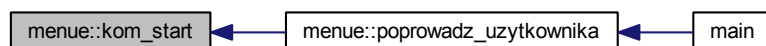
Metoda wyświetla komentarz na strumień wyjściowy.

Wyswietlany komentarz informuje, że zaraz rozpocznie się gra w życie oraz jak ją zakończyć. Metoda jest użyta tuż przed startem gry.

**Zwraca**

nic

Oto graf wywołań tej funkcji:





## 4.5.2.5 kom\_wpr\_danych()

```
void menue::kom_wpr_danych ( ) [static]
```

Metoda wyświetla komentarz na strumień wyjściowy.

Wyświetlany komentarz prosi użytkownika o wybór sposobu wprowadzenia współrzędnych komerek do ożywienia.

Zwraca

nic

Oto graf wywołań tej funkcji:



## 4.5.2.6 kom\_wpr\_danych\_w\_petli()

```
void menue::kom_wpr_danych_w_petli ( ) [static]
```

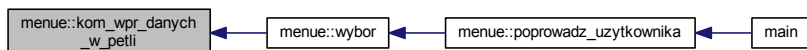
Metoda wyświetla komentarz na strumień wyjściowy.

Wyświetlany komentarz dotyczy wyboru sposobu wprowadzenia współrzędnych. Jest on wyświetlany w petli, która kończy się gdy użytkownik poda poprawną odpowiedź (1 lub 2).

Zwraca

nic

Oto graf wywołań tej funkcji:



#### 4.5.2.7 kom\_zapis()

```
void menue::kom_zapis ( ) [static]
```

Metoda wyswietla komentarz na strumien wyjsciowy.

Wyswietlany komentarz prosi uzytkownika o wybor, czy chce on zapisac aktualny stan(po zatrzymaniu) planszy do pliku.

Zwraca

nic

Oto graf wywoływań tej funkcji:



#### 4.5.2.8 kom\_zapis\_w\_petli()

```
void menue::kom_zapis_w_petli ( ) [static]
```

Metoda wyswietla komentarz na strumien wyjsciowy.

Wyswietlany komentarz dotyczy wyboru, czy uzytkownik chce zapisac plansze do pliku. Jest on wyswietlany w petli, która konczy sie gdy uzytkownik poda poprawna odpowiedz (1 lub 2).

Zwraca

nic

Oto graf wywoływań tej funkcji:



#### 4.5.2.9 petla()

```
void menue::petla (
    komorka ko,
    wyswietl wys ) [static]
```

Metoda zawiera nieskonczona petle do gry w zycie.

W metodzie, w nieskonczonej petli, wywoływane sa wszelkie metody konieczne do zmiany stanu planszy(ozywianie i zabijanie) oraz do wyswietlenia jej. Z petli da sie wyjsc przez zmiane wartosci pola statycznego klasy menue warunek\_wyjscia. Za zmiane tego pola odpowiedzialna jest zsynchronizowany watek metody zatrzymaj\_petle. Obiekty podane jako argumenty musialy byc kopiowane, poniewaz tak jest domyslnie w watku, który przyjmuje funkcje i jej rgumenty. Zwykla referencja nic nie zmieniala. Niestety nie potrafie jeszcze dodac do watku argumentow bez kopiowania i dzialac na oryginalach.

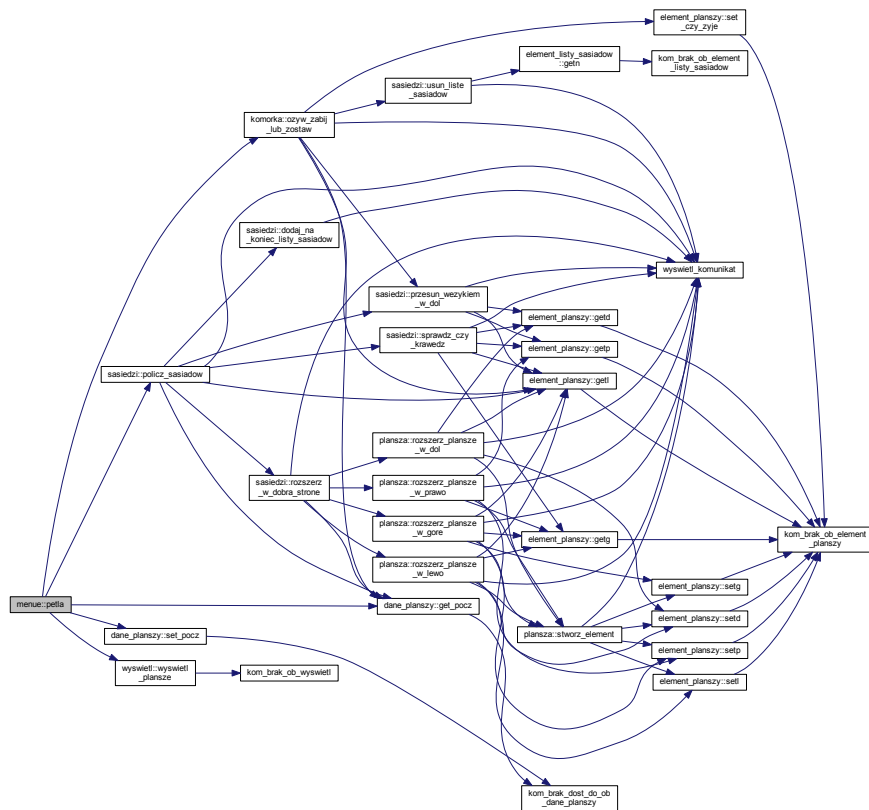
## Parametry

ko	kopia obiektu klasy komorka, za pomoca metod tego obiektu beda ozywiane i zabijane komorki
wys	kopia obiektu klasy wyswietl, za pomoca metody tego obiektu bedzie wyswietlana plansza do gry w zycie i jej aktualny stan.

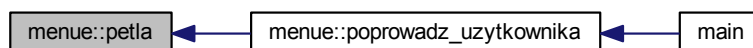
## Zwraca

nic

Oto graf wywołań dla tej funkcji:

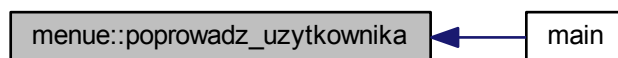


Oto graf wywoływań tej funkcji:





Oto graf wywołań tej funkcji:



#### 4.5.2.11 wybor()

```
int menue::wybor (
    int & zmiana_wyb ) [static]
```

Metoda do wyboru między dwoma opcjami.

Metoda zależna od parametru podanego jako argument wyświetla komunikaty związane z różnym wyborem. 1 - wybór sposobu wprowadzenia współrzędnych punktów do ożywienia 2 - wybór czy wystartować grę w życie, czy ją zakończyć 3 - wybór czy zapisać aktualny stan planszy po zatrzymaniu nieskończonej petli gry w życie. Metoda nie kończy się dopóki użytkownik nie poda liczby 1 lub 2 na wejście. Po zakończeniu zwraca nr wyboru, który później jest użyty do wywołania odpowiednich metod.

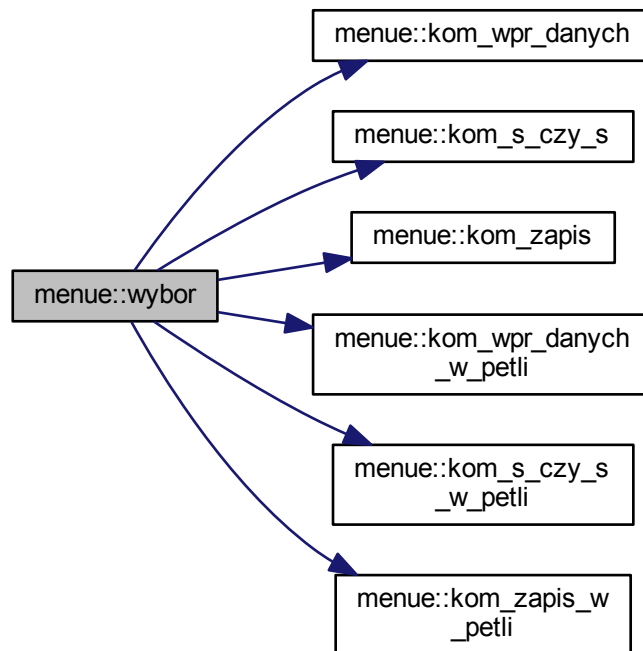
##### Parametry

<i>zmiana_wyboru</i>	jest indeksem wyboru, w zależności od jego wartości wyświetlają się inne opcje wyboru(1 - jak wprowadzić współrzędne, 2 - czy wystartować grę w życie, 3 - czy zapisać aktualny stan planszy).
----------------------	--

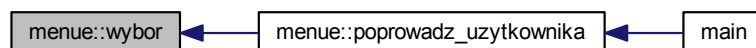
##### Zwraca

wybor - 48 (-48, ponieważ wybrane opcje zostają sczytane jako kod ascii)

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.5.2.12 zatrzymaj\_petle()

```
void menue::zatrzymaj_petle ( ) [static]
```

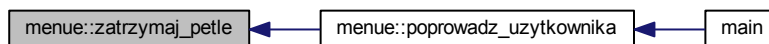
Metoda służy do zatrzymania nieskończonej petli do gry w życie.

Metoda czeka na podanie przez użytkownika danych na wejście. Jeśli podana dana jest litera n, metoda zmienia wartość pola klasy menue warunek\_wyjścia na true. Następnie po sprawdzeniu tego warunku w nieskończonej petli do gry w życie petla się zatrzymuje. Jeśli użytkownik podał coś innego, bufor się czyści i metoda jest wywołana ponownie rekurencyjnie. Rekurencja była konieczna, ponieważ inaczej metoda kończyła się przed metodą petla. Uniemożliwilo to wyjście z nieskończonej petli w sytuacji, gdy użytkownik najpierw podał niepoprawną daną (coś co nie było literą n). Wątek tej metody jest zsynchronizowany z wątkiem metody petla.

Zwraca

nic

Oto graf wywołań tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_menue.h](#)
- [metody\\_menue.cpp](#)

## 4.6 Dokumentacja klasy plansza

```
#include <klasa_plansza.h>
```

Diagram dziedziczenia dla plansza

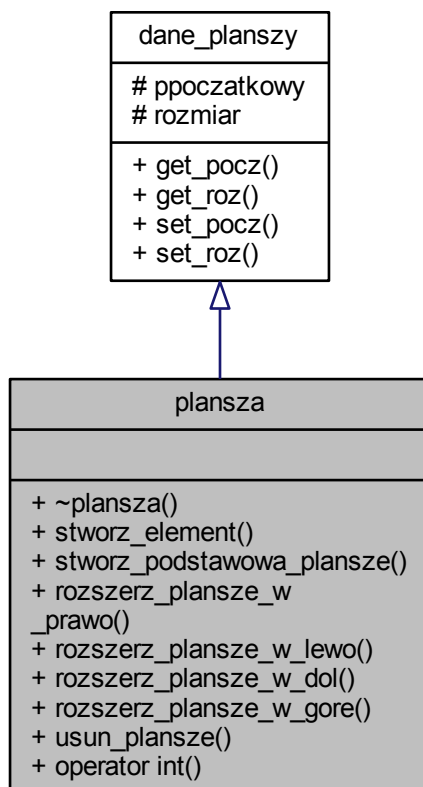
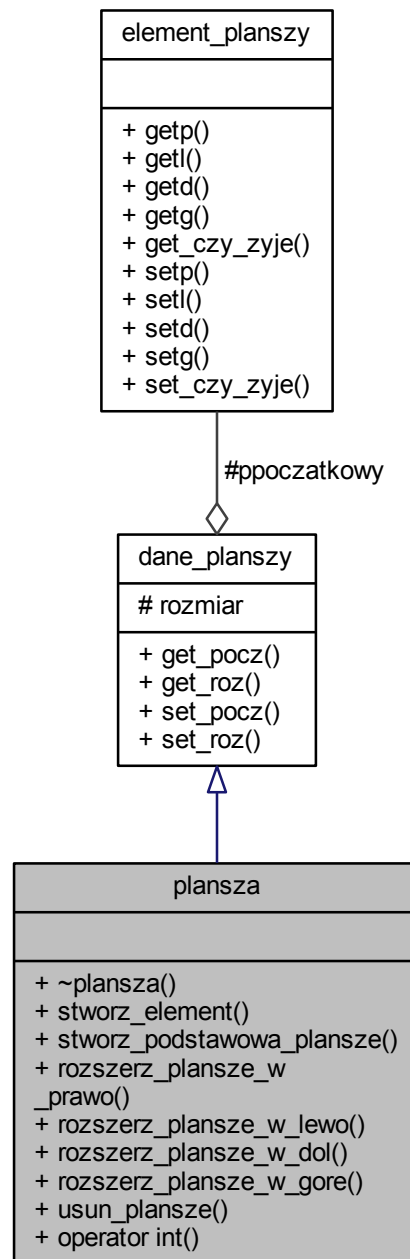


Diagram współpracy dla plansza:



## Metody publiczne

- `~plansza ()`
- `element_planszy * stworz_element (element_planszy *pp, element_planszy *pl, element_planszy *pd, element_planszy *pg)`
- `void stworz_podstawowa_plansze ()`
- `void rozszerz_plansze_w_prawo (element_planszy *el)`



- void `rozszerz_plansze_w_lewo` (`element_planszy *el`)
- void `rozszerz_plansze_w_dol` (`element_planszy *el`)
- void `rozszerz_plansze_w_gore` (`element_planszy *el`)
- void `usun_plansze` ()
- operator `int` ()

## Dodatkowe Dziedziczone Składowe

### 4.6.1 Opis szczegółowy

Klasa umożliwia operacje na planszy. Jest klasą pochodną klasy `dane_planszy`.

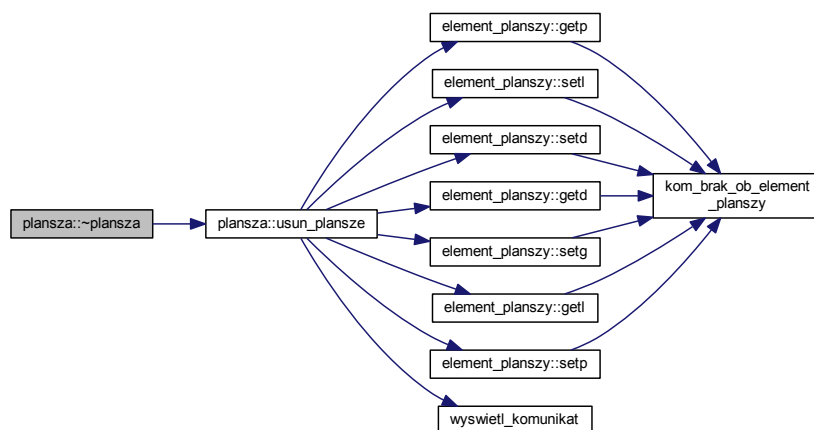
Klasa nie rozszerza klasy `dane_planszy` o żadne pole. Zawiera jedynie dodatkowe metody, takie jak tworzenie elementu planszy, tworzenie z nich podstawowej planszy, rozszerzanie planszy w daną stronę.

### 4.6.2 Dokumentacja konstruktora i destruktor

#### 4.6.2.1 `~plansza()`

```
plansza::~~plansza ( )
```

Destruktor klasy `plansza`. Poza usunięciem obiektu klasy `plansza` zwalnia pamięć zajmowaną przez elementy planszy do gry w życie. Jeśli początkowo na nic nie wskazuje destruktork tylko usuwa obiekt. Oto graf wywołań dla tej funkcji:



### 4.6.3 Dokumentacja funkcji składowych

#### 4.6.3.1 operator int()

```
plansza::operator int ( )
```

Przeciążony operator konwersji na int zlicza ilość elementów planszy. Nic nie pobiera i zwraca ilość elementów planszy.

Operator ustawia wskaźnik chwilowy tmp na adres wskazywany przez wskaźnik ppoczątkowy po czym przesuwa go w prawo zliczając kolumny. Następnie przesuwa ten sam chwilowy wskaźnik w dół i liczy wiersze. Ponieważ plansza ma zawsze kształt prostokątny wystarczy pomnożyć ilość kolumn razy ilość wierszy.

**Zwraca**

ilość elementów planszy

#### 4.6.3.2 rozszerz\_plansze\_w\_dol()

```
void plansza::rozszerz_plansze_w_dol (
    element_planszy * el )
```

Metoda rozszerza planszę o wiersz w dół. Pobiera wartość typu wskaźnikowego na obiekt klasy element planszy i nic nie zwraca.

Metoda jest używana do rozszerzenia planszy w sytuacji gdy komórka "ożyje" (pole bool czy\_zyje klasy `element_planszy` = 1) na dolnej krawędzi. Dzięki niej plansza jest nieskończona z dołu (ograniczona pamięcią komputera). Jeśli coś poszło nie tak i wskaźnik `el` wskazuje na `el` w środku planszy, metoda nic nie robi.

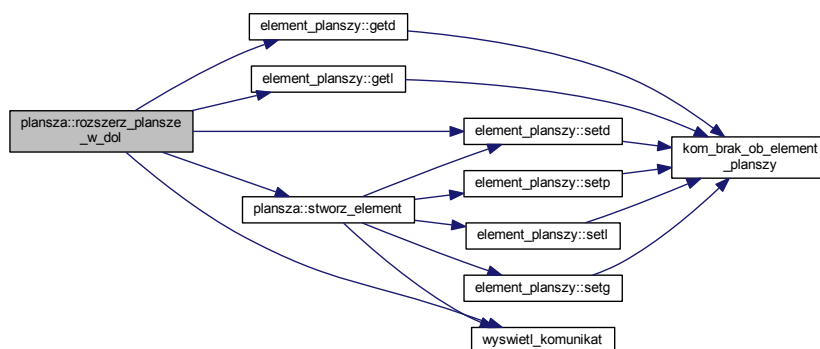
**Parametry**

<code>el</code>	wskaźnik na komórkę na dolnej krawędzi.
-----------------	---

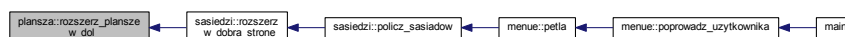
**Zwraca**

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



#### 4.6.3.3 rozszerz\_plansze\_w\_gore()

```
void plansza::rozszerz_plansze_w_gore (
    element_planszy * el )
```

Metoda rozszerza plansze o wiersz w gore. Pobiera wartosc typu wskaznikowego na obiekt klasy element planszy i nic nie zwraca.

Metoda jest uzywana do rozszerzenia planszy w sytuacji gdy komorka "ozyje"(pole bool czy\_zyje klasy `element_planszy` = 1) na gornej krawedzi. Dzieki niej plansza jest nieskonczona z gory (ograniczona pamiecia komputera). Jesli cos poszlo nie tak i wskaznik el wskazuje na el w srodku planszy, metoda nic nie robi. Metoda przesuwa wskaznik ppozatkowy z klasy `dane_planszy` na pierwszy element dodawanej kolumny(element w lewym gornym rogu).

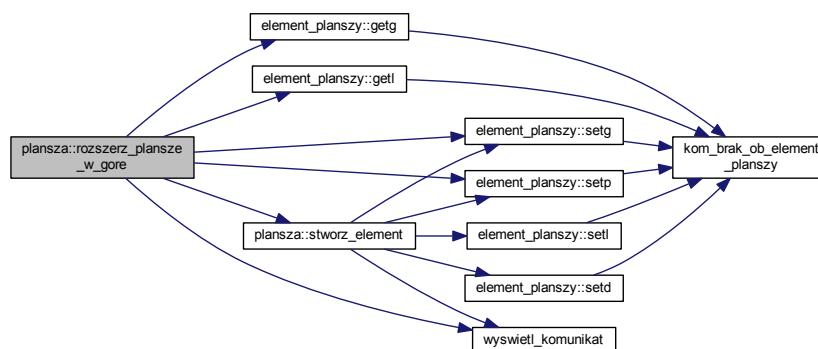
##### Parametry

<code>el</code>	wskaznik na komorce na gornej krawedzi.
-----------------	---

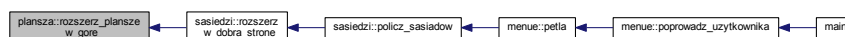
##### Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



#### 4.6.3.4 rozszerz\_plansze\_w\_lewo()

```
void plansza::rozszerz_plansze_w_lewo (
    element_planszy * el )
```

Metoda rozszerza plansze o kolumnę w lewo. Pobiera wartość typu wskaźnikowego na obiekt klasy element planszy i nic nie zwraca.

Metoda jest używana do rozszerzenia planszy w sytuacji gdy komórka "ożyje" (pole bool czy\_zyje klasy `element_planszy` = 1) na lewej krawędzi. Dzięki niej plansza jest nieskończona z lewej strony (ograniczona pamięcią komputera). Jeśli coś poszło nie tak i wskaźnik `el` wskazuje na `el` w środku planszy, metoda nic nie robi. Metoda przesuwa wskaźnik początkowy z klasy `dane_planszy` na pierwszy element dodawanej kolumny (element w lewym górnym rogu).

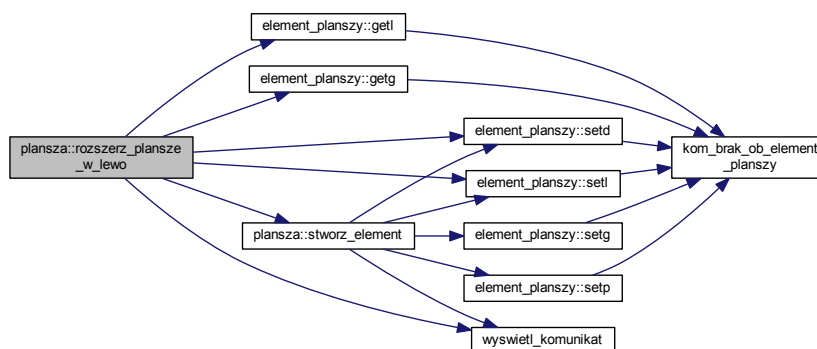
##### Parametry

<code>el</code>	wskaźnik na komórkę na lewej krawędzi.
-----------------	--

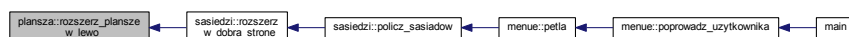
##### Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



#### 4.6.3.5 rozszerz\_plansze\_w\_prawo()

```
void plansza::rozszerz_plansze_w_prawo (
    element_planszy * el )
```

Metoda rozszerza plansze o kolumnę w prawo. Pobiera wartość typu wskaźnikowego na obiekt klasy `element` planszy i nic nie zwraca.

Metoda jest używana do rozszerzenia planszy w sytuacji gdy komórka "ożyje" (pole `bool` `czy_zyje` klasy `element_planszy` = 1) na prawej krawędzi. Dzięki niej plansza jest nieskończona z prawej strony (ograniczona pamięcią komputera). Jeśli coś poszło nie tak i wskaźnik `el` wskazuje na `el` w środku planszy, funkcja nic nie robi.

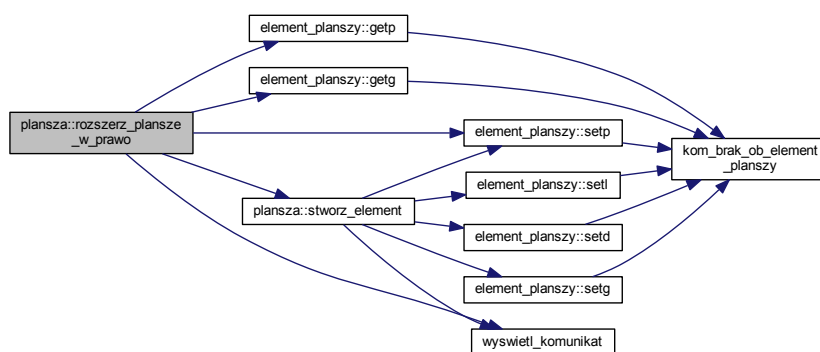
##### Parametry

<code>el</code>	wskaźnik na komórkę na prawej krawędzi.
-----------------	---

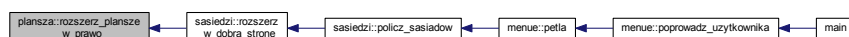
##### Zwraca

nowy - wskaźnik na nowo utworzony obiekt klasy `element_planszy`.

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



## 4.6.3.6 stworz\_element()

```

element_planszy * plansza::stworz_element (
    element_planszy * pp,
    element_planszy * pl,
    element_planszy * pd,
    element_planszy * pg )

```

Metoda tworzy element planszy, który jest reprezentacją komórki z gry w życie. Pobiera cztery wartości typu wskaźnikowego na obiekt klasy element planszy i zwraca również wskaźnik na obiekt klasy `element_planszy`.

W metodzie utworzony jest obiekt klasy `element_planszy` i ustawiony na niego wskaźnik nowy. Następnie ustawiane są wszystkie wskaźniki "sasiadow" na elementy podane w parametrach funkcji.

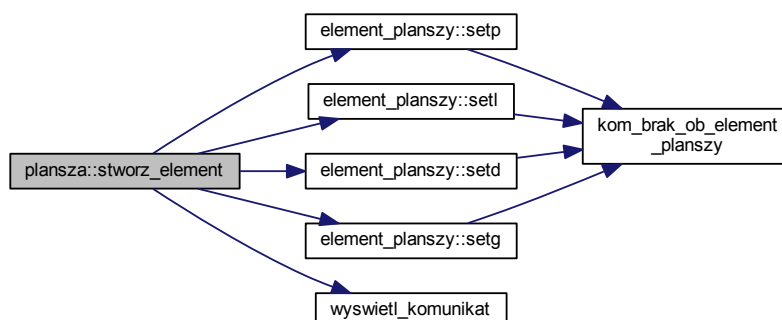
## Parametry

<i>pp</i>	wskaźnik na komórkę sąsiadującą po prawej lub na nullptr gdy tworzony element znajduje się na prawej krawędzi planszy
<i>pl</i>	wskaźnik na komórkę sąsiadującą po lewej lub na nullptr gdy tworzony element znajduje się na lewej krawędzi planszy
<i>pd</i>	wskaźnik na komórkę sąsiadującą z dołu lub na nullptr gdy tworzony element znajduje się na dolnej krawędzi planszy
<i>pg</i>	wskaźnik na komórkę sąsiadującą z góry lub na nullptr gdy tworzony element znajduje się na górnej krawędzi planszy

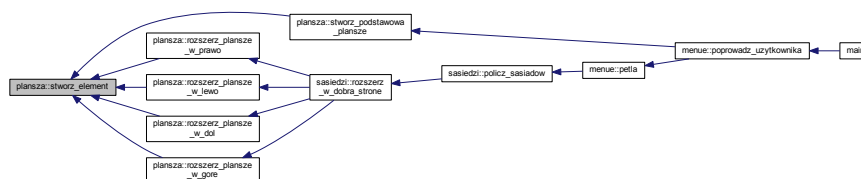
## Zwraca

nowy - wskaźnik na nowo utworzony obiekt klasy `element_planszy`.

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



#### 4.6.3.7 stworz\_podstawowa\_plansze()

```
void plansza::stworz_podstawowa_plansze ( )
```

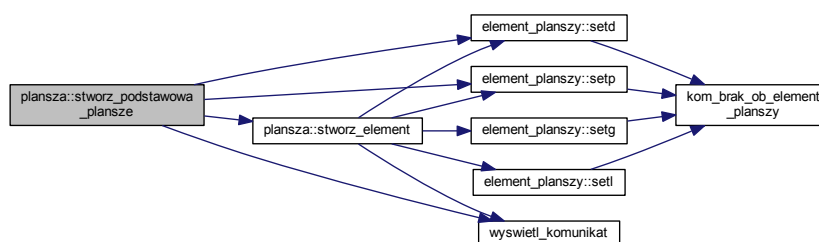
Metoda tworzy plansze o boku rownym polu klasy [dane\\_planszy](#) - rozmiar , na ktorej bedzie odbywala sie gra w zycie. Nie pobiera zadnych parametrow i nic nie zwraca.

Gdy pole rozmiar jest rowne 0 plansza w ogole sie nie tworzy. Jesli nie, funkcja najpierw tworzy glowna kolumne o dlugosci rozmiar, nastepnie tworzy kolejne kolumny, ktorych jest rowniez rozmiar. W metodzie wykorzystane sa trzy wskazniki chwilowe: tmp\_w\_dol - do przesuwania sie w dol po utworzonej lewej kolumnie i laczenia elementow wskaznikami pprawy z nowo tworzonymi. tmp\_prawy\_w\_dol - do przesuwania sie w dol i tworzenia nowej kolumny oraz laczenia jej z utworzona po lewej za pomoca wskaznikow plewy. tmp\_w\_prawo - do przesuniecie powyzzszych wskaznikow w prawo tak aby tmp\_w\_dol wskazywal na pierwszy el istniejacej kolumny, a tmp\_prawy\_w\_dol na pierwszy nowo utworzony w nastepnej.

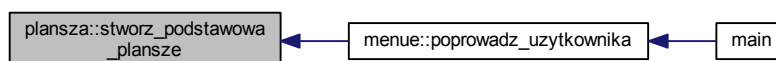
Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



## 4.6.3.8 usun\_plansze()

```
void plansza::usun_plansze ( )
```

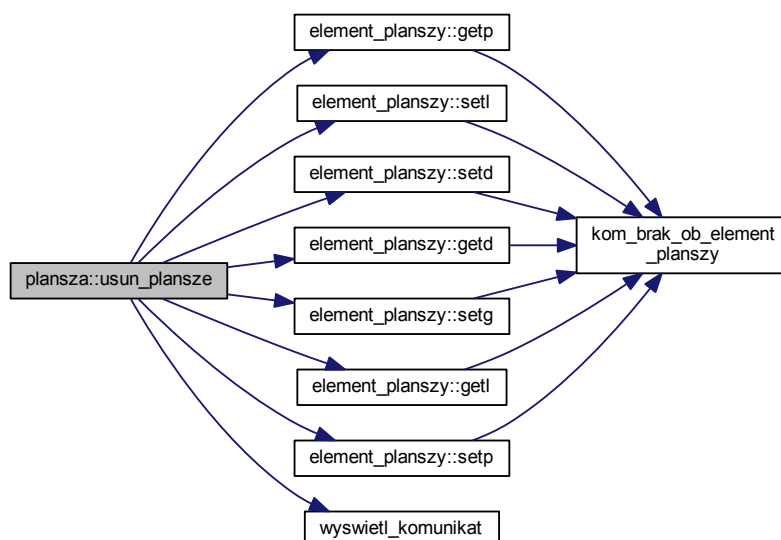
Metoda usuwa plansze do gry w zycie. Nic nie pobiera i nic nie zwraca.

Metoda ustawia wskaźnik chwilowy tmp na adres wskazywany przez wskaźnik ppoczątkowy po czym przesuwając wskaźnik ppoczątkowy w prawo. Następnie usuwa komórkę wskazywaną przez tmp oraz ustawia wskaźnik plewy komórki wskazywanej przez ppoczątkowy na nullptr. Kiedy wskaźnik ppoczątkowy "dojdzie" do prawej krawędzi planszy, jeśli to możliwe, przesuwa się w dół i wykonywana jest ten sam algorytm na następnym wierszu, tylko w lewo. Ostatni usunięty element ma wszystkie wskaźniki "sasiadów" ustawione na nullptr poza pgorny. Taka sytuacja spełnia instrukcję warunkową else if i ustawia wskaźnik ppoczątkowy na nullptr i pozwala na wyjście z petli while. Metoda została stworzona aby zapobiec wyciekowi pamięci. Jest użyta w destruktorze, dzięki czemu przy usuwaniu obiektu klasy plansza zostaje zwolniona pamięć zajmowana przez wszystkie elementy planszy.

Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_plansza.h](#)
- [metody\\_plansza.cpp](#)



## 4.7 Dokumentacja klasy sasiedzi

```
#include <klasa_sasiedzi.h>
```

Diagram dziedziczenia dla sasiedzi

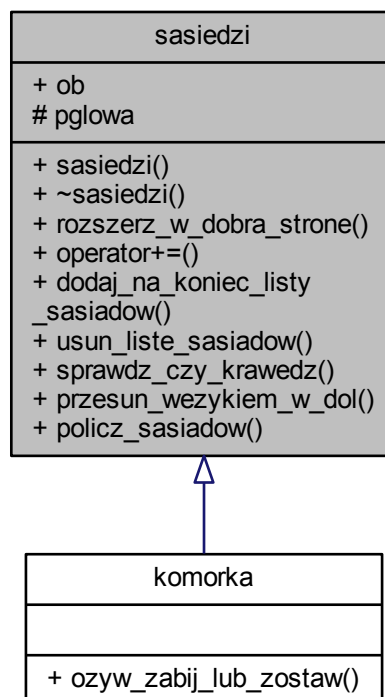
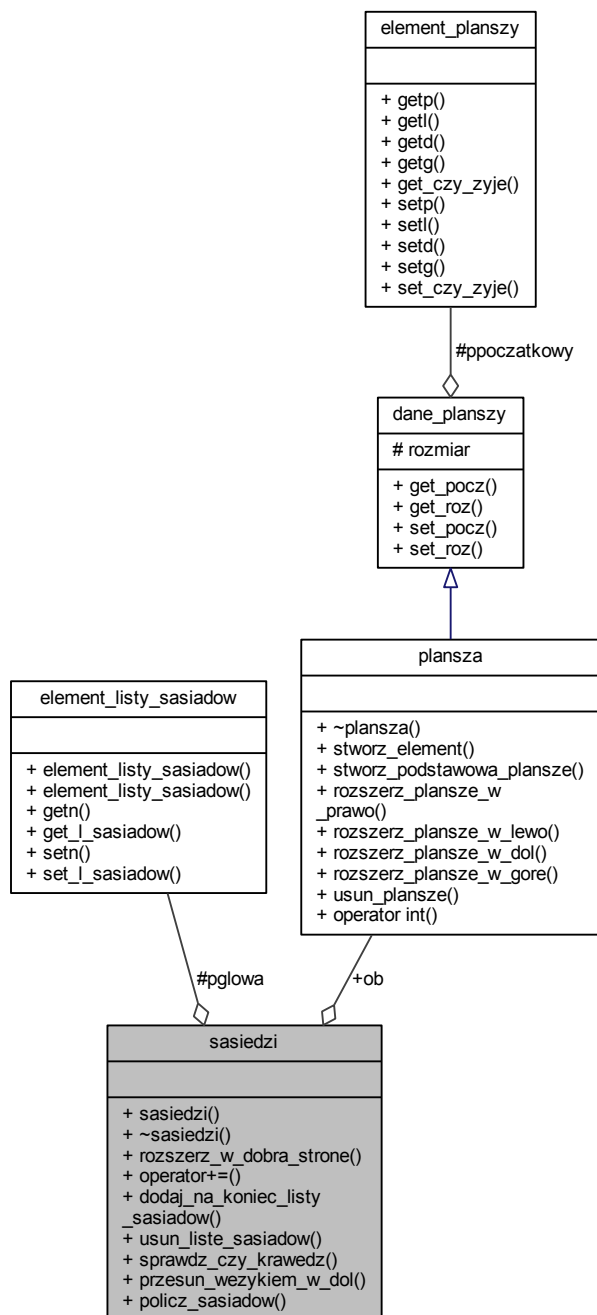


Diagram współpracy dla sasiedzi:



## Metody publiczne

- `sasiedzi ()`
- `~sasiedzi ()`
- `void rozszerz_w_dobra_strone ()`
- `element_listy_sasiadow * operator+= (int l_s)`
- `void dodaj_na_koniec_listy_sasiadow (int l_s)`

- void `usun_liste_sasiadow` ()
- int `sprawdz_czy_krawedz` (`element_planszy *el`)
- void `przesun_wezykiem_w_dol` (`element_planszy *&el`, `element_planszy *&el_z`)
- void `policz_sasiadow` ()

#### Atrybuty publiczne

- `plansza * ob`

#### Atrybuty chronione

- `element_listy_sasiadow * pglowa`

### 4.7.1 Opis szczegółowy

Klasa służy do zliczania sąsiadów komore planszy do gry w życie. Jest klasą bazową klasy komorka

### 4.7.2 Dokumentacja konstruktora i destruktora

#### 4.7.2.1 `sasiedzi()`

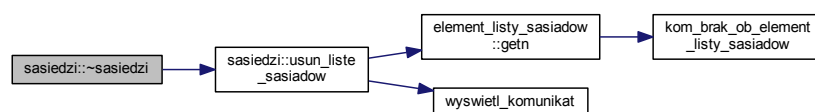
```
sasiedzi::sasiedzi ( )
```

Konstruktor klasy sasiedzi. Utworzony, aby ustawiać pole pglowa na nullptr przy tworzeniu nowego obiektu.

#### 4.7.2.2 `~sasiedzi()`

```
sasiedzi::~~sasiedzi ( )
```

Destruktor klasy sasiedzi. Utworzony, aby dodatkowo zwalniać pamięć zajmowaną przez listę sąsiadów (wywołana funkcja `usun_liste_sasiadow`). Jeśli pole pglowa na coś wskazuje, zostaje to usunięte. Oto graf wywołań dla tej funkcji:



### 4.7.3 Dokumentacja funkcji składowych

#### 4.7.3.1 `dodaj_na_koniec_listy_sasiadow()`

```
void sasiedzi::dodaj_na_koniec_listy_sasiadow (
    int l_s )
```

Metoda dodaje nowy element na koniec listy sąsiadów.

W metodzie użyty jest przeciążony operator arytmetyczny `+=`.

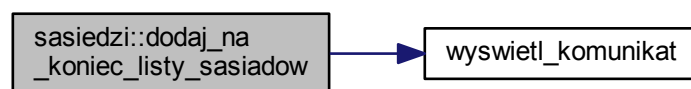
**Parametry**

$l \leftarrow$	liczba sasiadow komorki, ktora ma byc dodana do listy
$_ \leftarrow$	
$s$	

**Zwraca**

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

**4.7.3.2 operator+=()**

```

element_listy_sasiadow * sasiedzi::operator+= (
    int l_s )
  
```

Przeciazony operator arytmetyczny +=, ktory sluzi do dodawania elementow na koniec listy sasiadow.

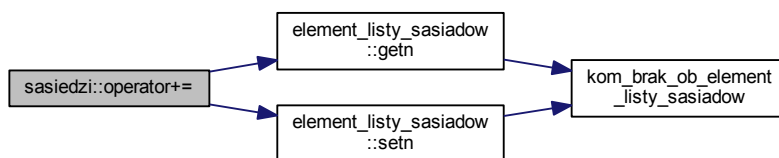
**Parametry**

$l \leftarrow$	liczba sasiadow komorki, ktora ma byc dodana do listy
$_ \leftarrow$	
$s$	

**Zwraca**

nowy przekształcony wskaźnik na pierwszy element listy sasiadow.

Oto graf wywołań dla tej funkcji:



#### 4.7.3.3 policz\_sasiadow()

```
void sasiedzi::policz_sasiadow ( )
```

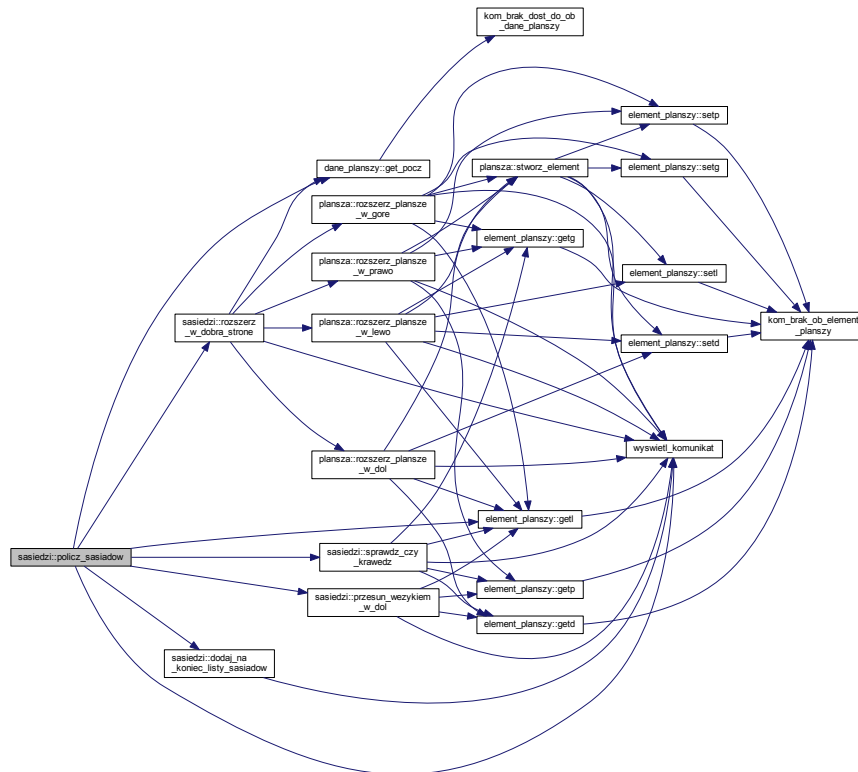
Metoda liczy sasiadow wszystkich komorek w planszy i dodaje ich liczbe na koniec listy sasiadow.

W metodzie wywołane są prawie wszystkie metody klasy `sasiedzi`. Na początku plansza jest rozszerzona w odpowiednie strony (metoda `rozszerz_w_dobra_strone`), aby móc sprawdzić ilość sąsiadów komorek, które wcześniej nie istniały. Zainicjowane są wskaźniki pomocnicze do przechodzenia planszy i liczenia sąsiadów. W pętli najpierw sprawdzana jest pozycja komórki (metoda `sprawdz_czy_krawedz`). Następnie w zależności od wartości zwróconej przez metodę `sprawdz_czy_krawedz` (czytaj opis tej metody) sprawdzana jest ilość sąsiadów. Ilość ta jest dodana na koniec listy sąsiadów (metoda `dodaj_na_koniec_listy_sasiadow`). Operacja powtarzana aż zostaną zliczeni sąsiedzi wszystkich komorek rozszerzonej planszy.

Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.7.3.4 przesun\_wezykiem\_w\_dol()

```
void sasiedzi::przesun_wezykiem_w_dol (
    element_planszy *& el,
    element_planszy *& el_za )
```

Metoda przesuwa wskaznik po planszy "wezykiem w dol".

Metoda przesuwa wskaznik podany jako parametr el o jedna pozycje. Najpierw do konca w prawo, jak juz sie nie da, raz w dol, potem w lewo. Przesuniecie jest mozliwe dopoki wskaznik na dolnego sasiada wskazuje na cos.

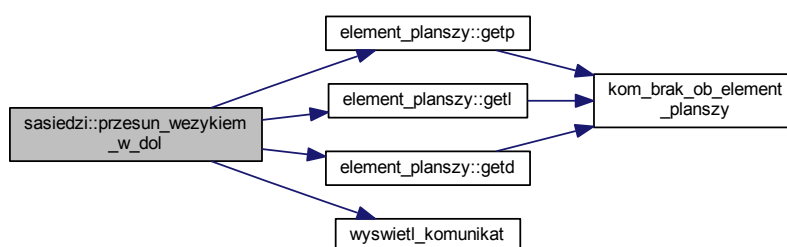
## Parametry

<i>el</i>	wskaznik na element planszy (do przesuniecie)
<i>el_za</i>	wskaznik na element za el w sensie przechodzenia wezykiem w dol(wskazuje na sasiada po lewej, jak metoda przesuwa w prawo, odwrotnie gdy w lewo)

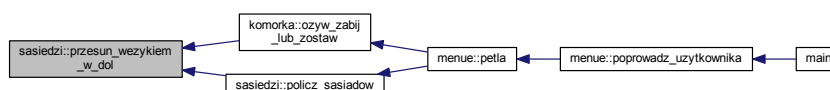
## Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



## 4.7.3.5 rozszerz\_w\_dobra\_strone()

```
void sasiedzi::rozszerz_w_dobra_strone ( )
```

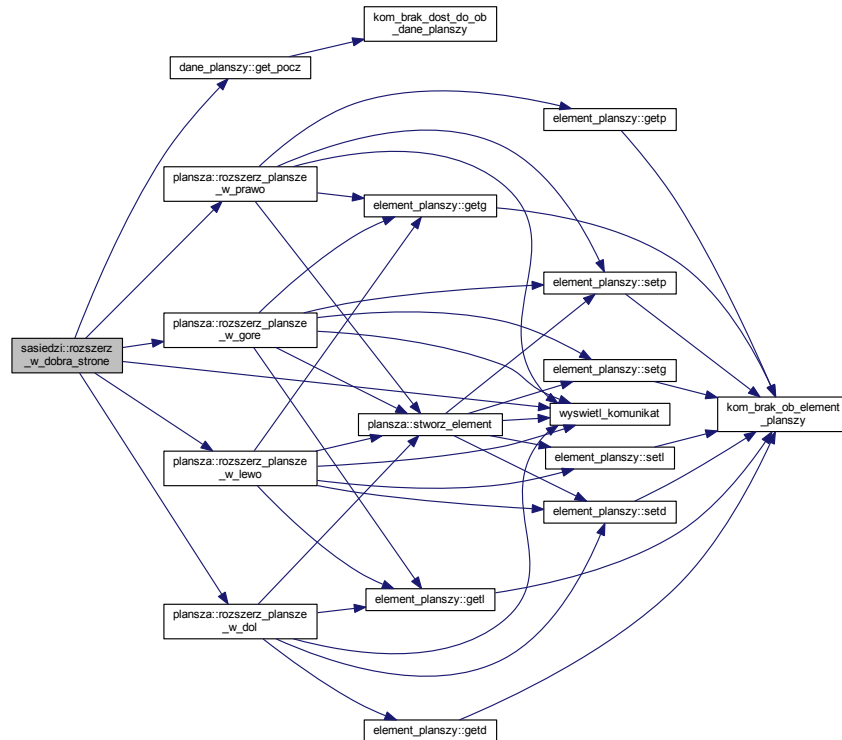
Metoda rozszerza plansze do gry w zycie w odpowiednia strone

Metoda uzywa pomocniczych wskaznikow i przesuwa jeden z nich wokol planszy sprawdzajac, czy na krawedzi jest jakas zywa komorka. Jesli tak, plansza jest rozszerzana(wywolana jest odpowiednia metoda klasy plansza) w strone krawedzi przy ktorej znaleziona zostala zywa komorka.

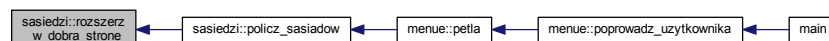
Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.7.3.6 sprawdź czy krawędź()

```
int sasiedzi::sprawdz_czy_krawedz (
    element_planszy * el )
```

Metoda sprawdza położenie komórki i w zależności od niego zwraca odpowiednią liczbę.

Znaczenie zwracanej liczby, to lokalizacja komórki w: 0 - prawa krawędź 10 - prawy dolny róg 20 - prawy górny róg 1 - lewa krawędź 11 - lewy dolny róg 21 - lewy górny róg 2 - dolna krawędź 3 - gorna krawędź Dzięki tej metodzie metoda policz\_sasiadow nie sprawdza sąsiadów, którzy nie mają prawa być żywymi.



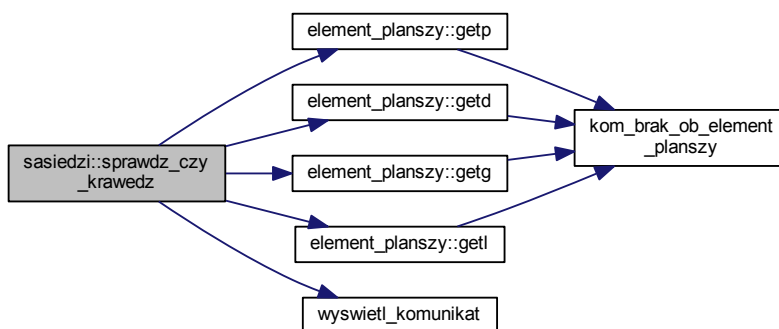
## Parametry

<i>e/</i>	wskaznik na komorke, ktorej lokalizacje chcemy poznać.
-----------	--

## Zwraca

liczba odpowiadajaca lokalizacji komorki.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



## 4.7.3.7 usun\_liste\_sasiadow()

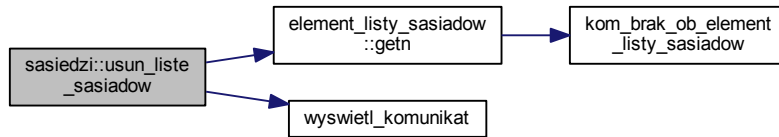
```
void sasiedzi::usun_liste_sasiadow ( )
```

Metoda usuwa i zwalnia pamiec po liscie sasiadow. Metoda wywolana w destruktorze.

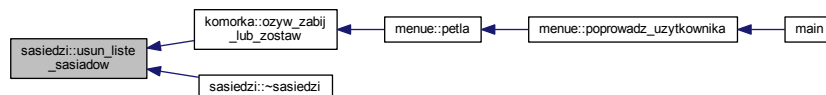
Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



## 4.7.4 Dokumentacja atrybutów składowych

### 4.7.4.1 ob

```
plansza* sasiedzi::ob
```

Pole ob jest wskaźnikiem na obiekt klasy plansza. Takie rozwiązanie zostało zastosowane zamiast dziedziczenia po klasie [dane\\_planszy](#), aby nie tworzyć tymczasowego obiektu klasy plansza do wywołania metod do rozszerzania planszy.

### 4.7.4.2 pglowa

```
element_listy_sasiadow* sasiedzi::pglowa [protected]
```

Pole pglowa jest wskaźnikiem na pierwszy element listy sasiadow.

Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_sasiedzi.h](#)
- [metody\\_sasiedzi.cpp](#)

## 4.8 Dokumentacja klasy wspolrzedne

```
#include <klasa_wspolrzedne.h>
```

Diagram dziedziczenia dla wspolrzedne

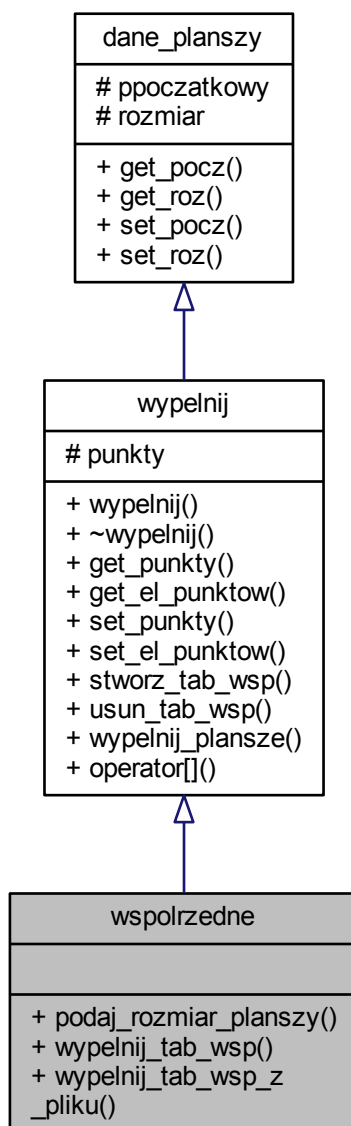
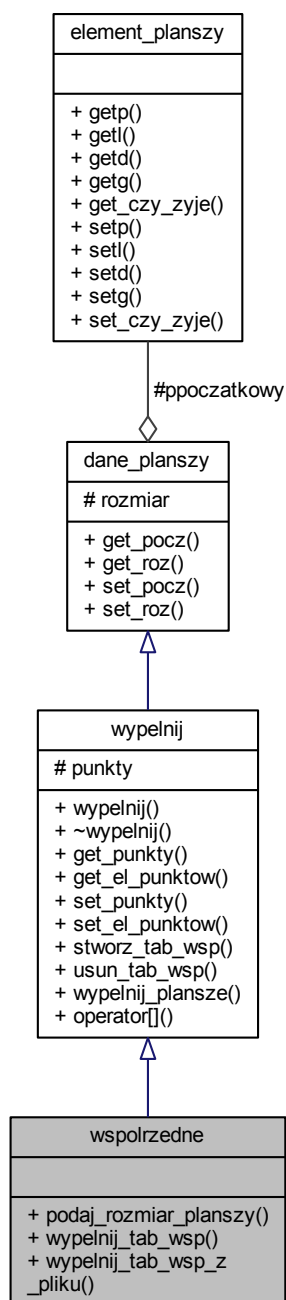


Diagram współpracy dla współrzędne:



## Metody publiczne

- void [podaj\\_rozmiar\\_planszy \(\)](#)
- void [wypelnij\\_tab\\_wsp \(\)](#)
- void [wypelnij\\_tab\\_wsp\\_z\\_pliku \(\)](#)

## Dodatkowe Dziedziczone Składowe

### 4.8.1 Opis szczegółowy

Klasa służy do zaczytywania współrzędnych komorek do ożywienia (na planszy do gry w życie) do tablicy dynamicznej punkty (pole klasy wypełnij). Jest klasa pochodna klasy [wypełnij](#) (która jest klasą pochodną klasy [dane\\_planszy](#)).

Klasa została stworzona aby oddzielić operacje wczytania i wyświetlania komunikatów związanych z tym od logiki metod klasy sąsiedzi. Metody tej klasy zawierają również operacje zapobiegające wpisaniu niepoprawnych danych. Wykorzystana jest maszyna stanów. Zawiera dodatkowo typ wyliczeniowy oraz metody.

### 4.8.2 Dokumentacja funkcji składowych

#### 4.8.2.1 podaj\_rozmiar\_planszy()

```
void wspolrzedne::podaj_rozmiar_planszy ( )
```

Metoda służy do wprowadzenia rozmiaru kwadratowej planszy początkowej (długość boku).

Wyświetlany jest komunikat, który prosi użytkownika o podanie rozmiaru planszy, tak aby zmieściły się wszystkie komórki do ożywienia. Następnie użytkownik powinien podać ten rozmiar. Operacja się powtarza dopóki użytkownik nie poda liczby naturalnej. Każdorazowo przy wpisaniu niepoprawnych danych wyświetlany jest odpowiedni komunikat.

Zwraca

nic

Oto graf wywołań tej funkcji:



#### 4.8.2.2 wypelnij\_tab\_wsp()

```
void wspolrzedne::wypelnij_tab_wsp ( )
```

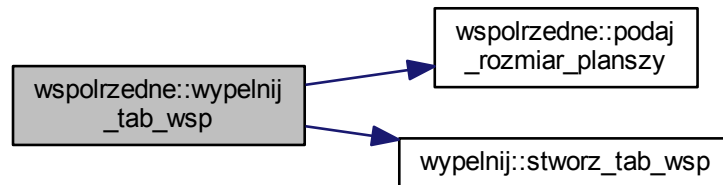
Metoda sluzy do wypelnienia tablicy dynamicznej punkty(pole klasy wypelnij) przez uzytkownika(recznie).

Najpierw sczytany jest rozmiar kwadratowej planszy poczatkowej(metoda podaj\_rozmiar\_planszy). Tworzona jest tablica wspolrzednych punkty(metoda klasy wypelnij stworz\_tab\_wsp) o rozmiarze rozmiar \* rozmiar \* 2 (jest to ilosc komorek na planszy \* 2(dwie wspolrzedne)). Wyszylony jest komunikat jak zakonczyc wpisywanie wspolrzednych komorek do ozywienia. W petli wykorzystana jest maszyna stanow do pobrania od uzytkownika wspolrzednych. Wyszylany komentarz prosi uzytkownika o wybor sposobu wprowadzenia wspolrzednych komorek do ozywienia.

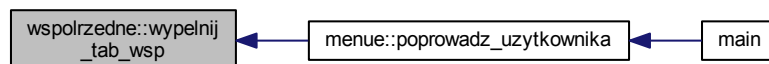
Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



#### 4.8.2.3 wypelnij\_tab\_wsp\_z\_pliku()

```
void wspolrzedne::wypelnij_tab_wsp_z_pliku ( )
```

Metoda sluzy do wypelnienia tablicy dynamicznej punkty(pole klasy wypelnij) z pliku o podanej nazwie.

Najpierw zostaje wyswietlony komunikat proszacy o podanie nazwy pliku i wyjasniajacy jak beda interpretowane dane w pliku. Po podaniu nazwy otwierany jest plik. Wykorzystany przeciazony operator strumieniowy >>. Jezeli plik nie istnieje, wyrzucany jest wyjatke przez ten operator. Jesli istnieje, dane zostaja wczytane z pliku w podany w komunikacie sposob.

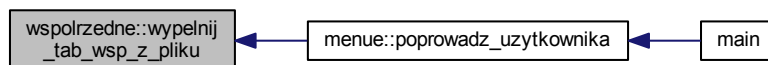
Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_wspolzedne.h](#)
- [metody\\_wspolzedne.cpp](#)

## 4.9 Dokumentacja klasy wypelnij

```
#include <klasa_wypelnij.h>
```

Diagram dziedziczenia dla wypelnij

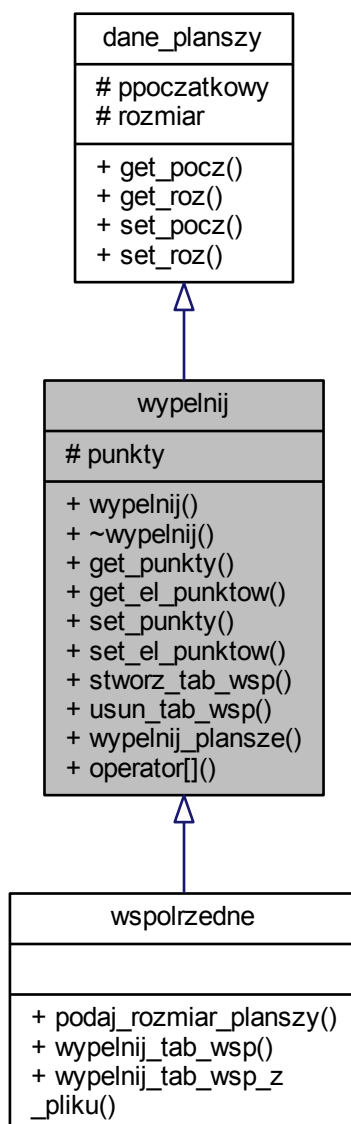
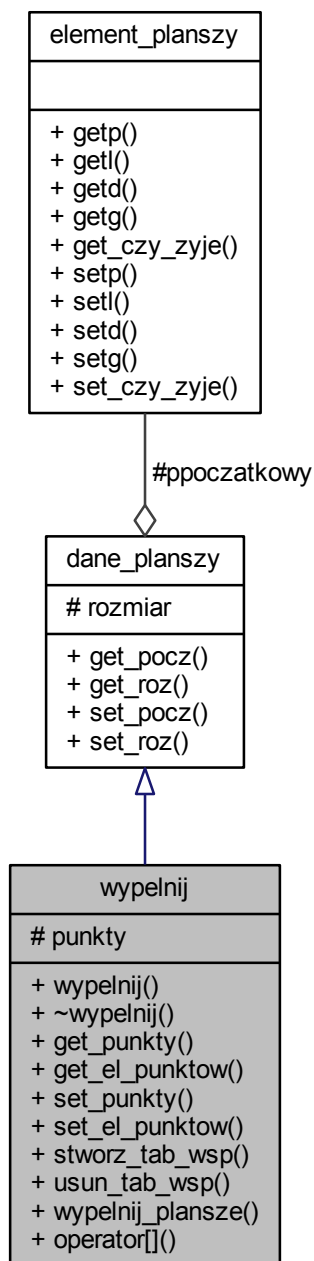




Diagram współpracy dla wypelnij:



### Metody publiczne

- `wypelnij ()`
- `~wypelnij ()`
- `int * get_punkty ()`
- `int get_el_punktow (int idx)`
- `void set_punkty (int *p)`

- void `set_el_punktow` (int idx, int liczba)
- void `stworz_tab_wsp` ()
- void `usun_tab_wsp` ()
- void `wypelnij_plansze` ()
- `element_planszy * operator[]` (size\_t el)

### Atrybuty chronione

- int \* `punkty`

## 4.9.1 Opis szczegółowy

Klasa służy do wypełniania planszy do gry w życie żywymi komórkami. Jest klasa bazowa klasy współrzędne.

## 4.9.2 Dokumentacja konstruktora i destruktora

### 4.9.2.1 `wypelnij()`

```
wypelnij::wypelnij ( )
```

Konstruktor klasy `wypelnij`. Utworzony, aby ustawiać pole `punkty` na `nullptr` przy tworzeniu nowego obiektu.

### 4.9.2.2 `~wypelnij()`

```
wypelnij::~~wypelnij ( )
```

Destruktor klasy `wypelnij`. Utworzony, aby dodatkowo zwalniać pamięć zajmowaną przez tablice dynamiczne `punkty` (wywołana funkcja `usun_tab_wsp`). Jeśli pole `punkty` na coś wskazuje, zostaje to usunięte. Oto graf wywołań dla tej funkcji:



## 4.9.3 Dokumentacja funkcji składowych

### 4.9.3.1 `get_el_punktow()`

```
int wypelnij::get_el_punktow (
    int idx )
```

Metoda zwraca element tablicy dynamicznej `punkty` o podanym w argumencie indeksie.

Metoda została stworzona aby uzyskać dostęp do jednego elementu pola `punkty` (wskazanego przez argument) przez klasy niezaprzyjżaznione i niepo pochodne. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy `wypelnij` (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

## Parametry

<i>idx</i>	indeks elementu tablicy dynamicznej do zwrocenia
------------	--

## Zwraca

punkty, jesli jest dotep do obiektu klasy wypelnij. Jesli nie, zwraca -1.

Oto graf wywołań dla tej funkcji:

4.9.3.2 `get_punkty()`

```
int * wypelnij::get_punkty ( )
```

Metoda zwraca pole punkty.

Metoda została stworzona aby uzyskac dostep do pola punkty przez klasy niezaprzyjznione i nie pochodne. Jesli cos poszlo nie tak i nie ma dostepu do obiektu klasy wypelnij (lub obiektu klasy pochodnej), metoda wyrzuca wyjatek.

## Zwraca

punkty, jesli jest dotep do obiektu klasy [dane\\_planszy](#). Jesli nie, zwraca nullptr.

Oto graf wywołań dla tej funkcji:



#### 4.9.3.3 operator[]()

```
element_planszy * wypelnij::operator[] (
    size_t el )
```

Przeciazony operator tablicowy [], który ustawia pomocniczy wskaźnik na elemencie o podanym w argumencie indeksie i zwraca go.

Operator ustawia wskaźnik chwilowy tmp na adres elementu planszy do gry w życie o podanym przez argument indeksie. Indeks jest interpretowany w następujący sposób: Ponieważ metoda przesuwa pomocniczy wskaźnik kolejno po wierszach, indeks oznacza liczbę przesunień konieczną do ustawienia tego wskaźnika. Indeks równy jest pierwszej współrzędnej  $x$  + rozmiar \* druga współrzędna  $y$ . Jeśli okaże się, że współrzędna w tablicy punkty wskazywała na element poza planszą zostaje wyrzucony wyjątek.

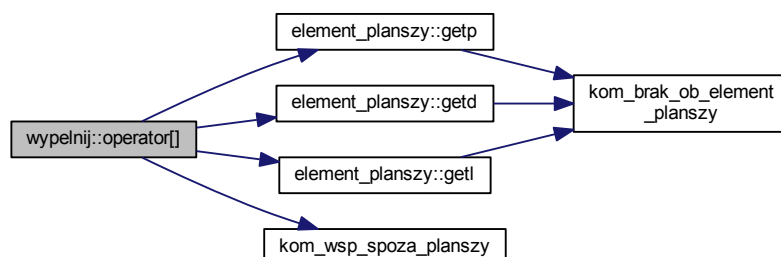
##### Parametry

<i>el</i>	indeks komórki do ozywienia (ilosc koniecznych przesuniec od elementu początkowego, aby dostac sie do pozodanej komórki)
-----------	--

##### Zwraca

adres na który ustawiony jest pomocniczy wskaźnik, czyli elementu o podanym w argumencie indeksie.

Oto graf wywołań dla tej funkcji:



#### 4.9.3.4 set\_el\_punktow()

```
void wypelnij::set_el_punktow (
    int idx,
    int liczba )
```

Metoda ustawia wartość jednego elementu tablicy dynamicznej punkty. Wartość podana jako drugi argument wpisana jest do elementu tablicy punkty o podanym indeksie w pierwszym argumencie.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę wartości elementu tablicy punkty (wskazanego przez arg idx) przez klasy niezaprzyjżnione i nie pochodne. Do pola przypisywana jest wartość przekazana przez drugi argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy wypelnij (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

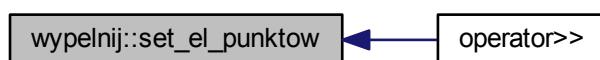
## Parametry

<i>idx</i>	indeks elementu tablicy dynamicznej do zmiany wartosci.
<i>liczba</i>	wartosc , ktora ma byc przypisana elementowi tablicy punkty.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



## 4.9.3.5 set\_punkty()

```
void wypelnij::set_punkty (  
    int * p )
```

Metoda ustawia pole punkty.

Metoda została stworzona aby uzyskać dostęp i pozwolić na zmianę wartości pola punkty przez klasy niezaprzyjane i niepo pochodne. Do pola przypisywana jest wartość przekazana przez argument. Jeśli coś poszło nie tak i nie ma dostępu do obiektu klasy wypelnij (lub obiektu klasy pochodnej), metoda wyrzuca wyjątek.

## Parametry

<i>p</i>	wskaznik na tablice dynamiczna, na ktora ma wskazywac pole punkty.
----------	--

Oto graf wywołań dla tej funkcji:



#### 4.9.3.6 stworz\_tab\_wsp()

```
void wypelnij::stworz_tab_wsp ( )
```

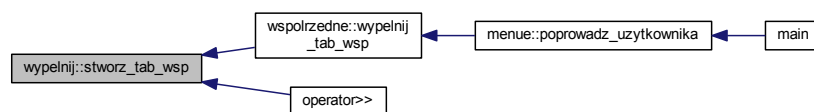
Metoda tworzy tablice dynamiczna punkty o rozmiarze  $2 * \text{rozmiar} * \text{rozmiar}$  [jest to ilosc komorek na planszy \* 2(dwie wspolrzedne)] oraz wypelnia ja wartosciami -1.

Rozmiar stworzonej tablicy odpowia liczbie elementow planszy \* 2. Mozna wiec maksymalnie wpisac tyle elementow, zeby zaplenic cala stworzona plansze.

**Zwraca**

nic

Oto graf wywoływań tej funkcji:



#### 4.9.3.7 usun\_tab\_wsp()

```
void wypelnij::usun_tab_wsp ( )
```

Metoda usuwa i zwalnia pamiec po tablicy dynamicznej punkty. Metoda zabezpieczona przed kilkukrotnym usunieciem Metoda wywolana w destruktorze.

Zwraca

nic

Oto graf wywołań tej funkcji:



#### 4.9.3.8 wypelnij\_plansze()

```
void wypelnij::wypelnij_plansze ( )
```

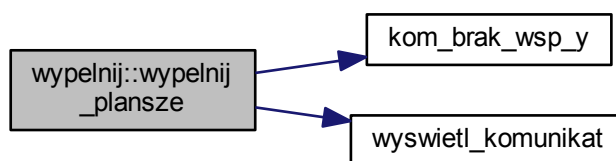
Metoda sluzy do wypelnienia tablicy dynamicznej punkty.

Metoda przesuwa wskaznik pomocniczy na element planszy wskazany przez wspolrzedne w tablicy punkty. Następnie ozywia ten element (zmiana pola klasy `element_planszy` czy\_zyje na true). Metoda wykorzystuje przeciazony operator tablicowy `[]` do przesuniecie wskaznika pomocniczego na odpowiedni element.

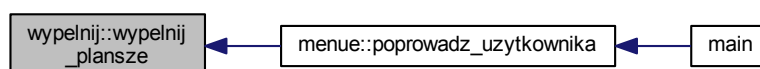
Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



#### 4.9.4 Dokumentacja atrybutów składowych

##### 4.9.4.1 punkty

```
int* wypelnij::punkty [protected]
```

Pole wskaźnikowe na int, służy do tworzenia tablicy dynamicznej z współzrzednymi punktów do ożywienia.

Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_wypelnij.h](#)
- [metody\\_wypelnij.cpp](#)

#### 4.10 Dokumentacja klasy wyswietl

```
#include <klasa_wyswietl.h>
```

Diagram dziedziczenia dla wyswietl

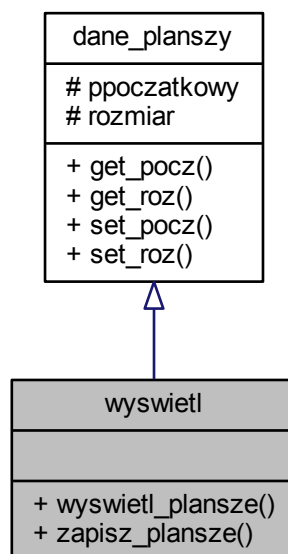
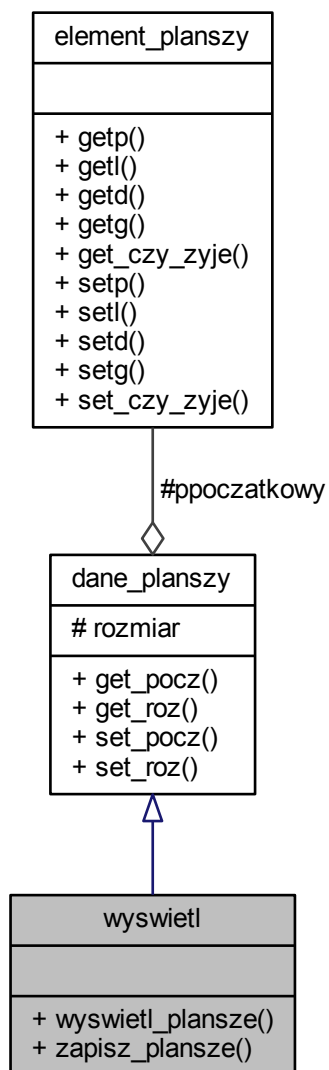




Diagram współpracy dla wyswietl:



### Metody publiczne

- void [wyswietl\\_plansze](#) ()
- void [zapisz\\_plansze](#) ()

### Dodatkowe Dziedziczone Składowe

#### 4.10.1 Opis szczegółowy

Klasa służy do wyświetlania oraz wpisywania aktualnego stanu planszy do pliku. Jest klasa pochodna klasy [dane\\_planszy](#). Rozszerza tylko o metody.

## 4.10.2 Dokumentacja funkcji składowych

### 4.10.2.1 wyswietl\_plansze()

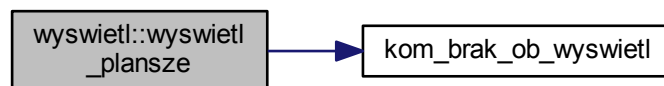
```
void wyswietl::wyswietl_plansze ( )
```

Metoda wyswietla aktualny stan planszy na strumien wyjsciowy. Metoda wykorzystuje przeciazony operator strumieniowy <<.

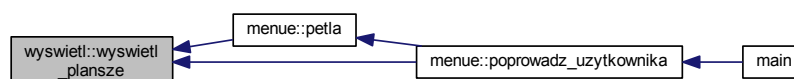
**Zwraca**

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



### 4.10.2.2 zapisz\_plansze()

```
void wyswietl::zapisz_plansze ( )
```

Metoda zapisuje aktualny stan planszy do pliku o podanej przez uzytkownika nazwie.

Najpierw zostaje wyswietlony komunikat proszacy o podanie nazwy pliku. Po podaniu nazwy otwierany jest plik. Wykorzystany przeciazony operator strumieniowy <<. Jezeli plik nie istnieje, wyrzucany jest wyjatek przez ten operator. Jesli istnieje dane zostaja wpisane do pliku.

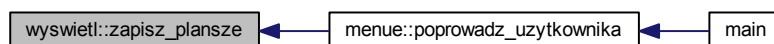
Zwraca

nic

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- [klasa\\_wyswietl.h](#)
- [metody\\_wyswietl.cpp](#)



## Rozdział 5

# Dokumentacja plików

### 5.1 Dokumentacja pliku klasa\_dane\_planszy.h

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

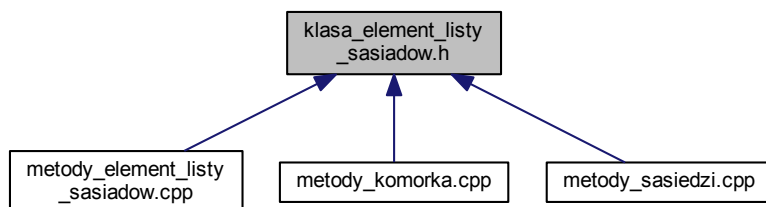


#### Komponenty

- class `dane_planszy`

### 5.2 Dokumentacja pliku klasa\_element\_listy\_sasiadow.h

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

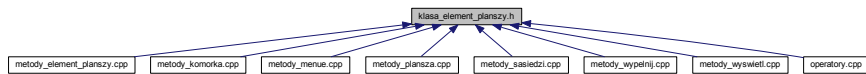


#### Komponenty

- class `element_listy_sasiadow`

### 5.3 Dokumentacja pliku `klasa_element_planszy.h`

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

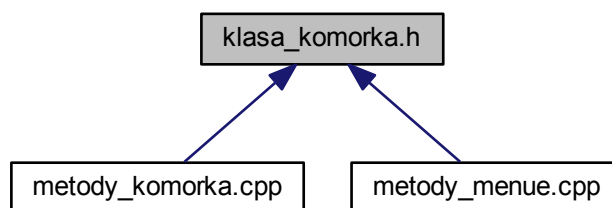


#### Komponenty

- class `element_planszy`

### 5.4 Dokumentacja pliku `klasa_komorka.h`

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

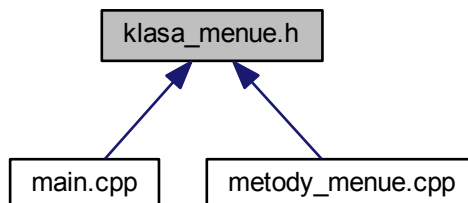


#### Komponenty

- class `komorka`

### 5.5 Dokumentacja pliku `klasa_menuue.h`

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

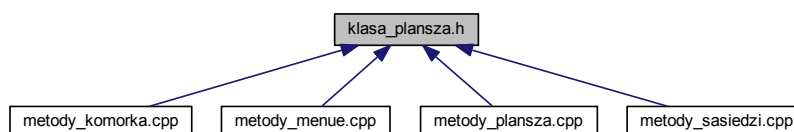


## Komponenty

- class [menue](#)

## 5.6 Dokumentacja pliku klasa\_plansza.h

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

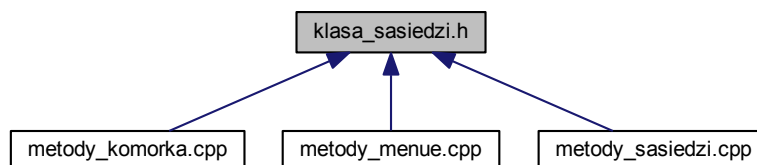


## Komponenty

- class [plansza](#)

## 5.7 Dokumentacja pliku klasa\_sasiedzi.h

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

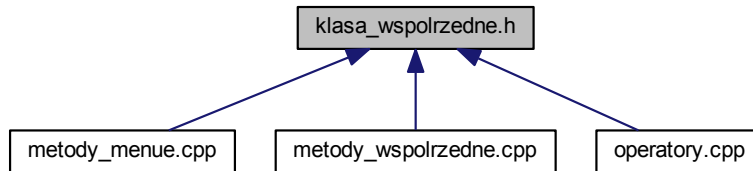


## Komponenty

- class [sasiedzi](#)

## 5.8 Dokumentacja pliku klasa\_wspolrzedne.h

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

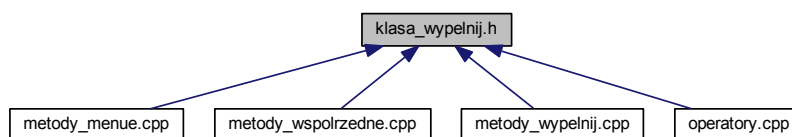


### Komponenty

- class [wspolrzedne](#)

## 5.9 Dokumentacja pliku klasa\_wypelnij.h

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

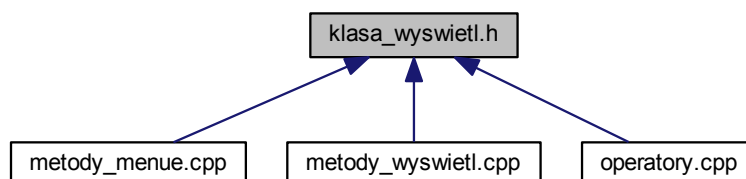


### Komponenty

- class [wypelnij](#)

## 5.10 Dokumentacja pliku klasa\_wyswietl.h

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





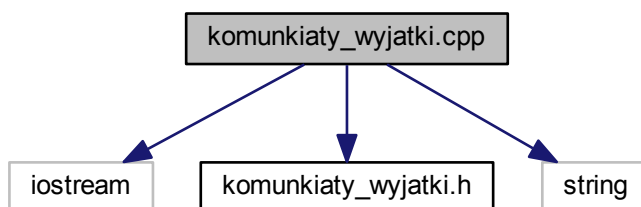
## Komponenty

- class `wyswietl`

## 5.11 Dokumentacja pliku komunikaty\_wyjatki.cpp

```
#include <iostream>
#include "komunikaty_wyjatki.h"
#include <string>
```

Wykres zależności załączania dla komunikaty\_wyjatki.cpp:



## Funkcje

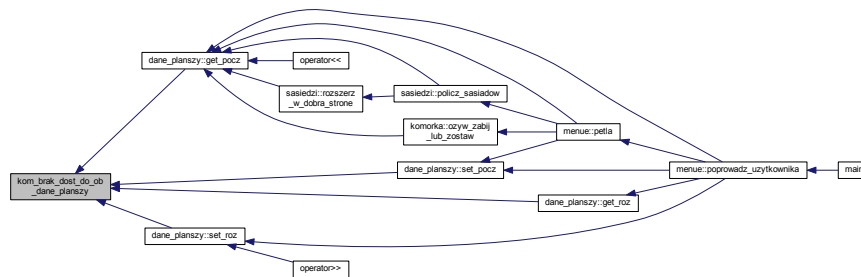
- void `wyswietl_komunikat` (std::string str)
- std::string `kom_brak_dost_do_ob_dane_planszy` ()
- std::string `kom_brak_ob_element_planszy` ()
- std::string `kom_brak_ob_wypelnij` ()
- std::string `kom_wsp_spoza_planszy` ()
- std::string `kom_brak_wsp_y` ()
- std::string `kom_brak_ob_wyswietl` ()
- std::string `kom_brak_pliku` ()
- std::string `kom_brak_ob_element_listy_sasiadow` ()

### 5.11.1 Dokumentacja funkcji

### 5.11.1.1 kom\_brak\_dost\_do\_ob\_dane\_planszy()

```
std::string kom_brak_dost_do_ob_dane_planszy ( )
```

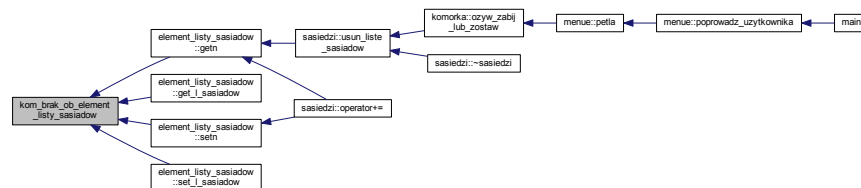
Funkcja zwraca napis informujący o wyjątku w klasie [dane\\_planszy](#). Napis informuje o braku dostępu do obiektu klasy [dane\\_planszy](#). Oto graf wywołań tej funkcji:



### 5.11.1.2 kom\_brak\_ob\_element\_listy\_sasiadow()

```
std::string kom_brak_ob_element_listy_sasiadow ( )
```

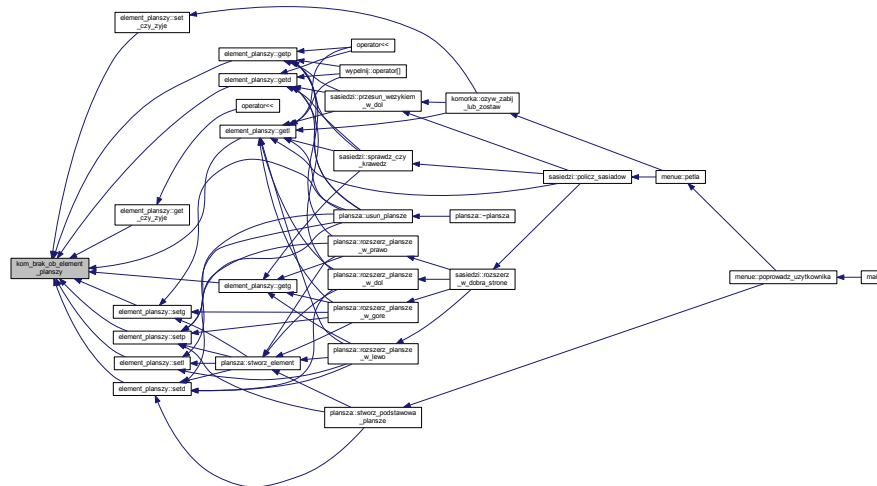
Funkcja zwraca napis informujący o wyjątku w klasie [element\\_listy\\_sasiadow](#). Napis informuje o braku dostępu do obiektu klasy [element\\_listy\\_sasiadow](#). Oto graf wywołań tej funkcji:



### 5.11.1.3 kom\_brak\_ob\_element\_planszy()

```
std::string kom_brak_ob_element_planszy ( )
```

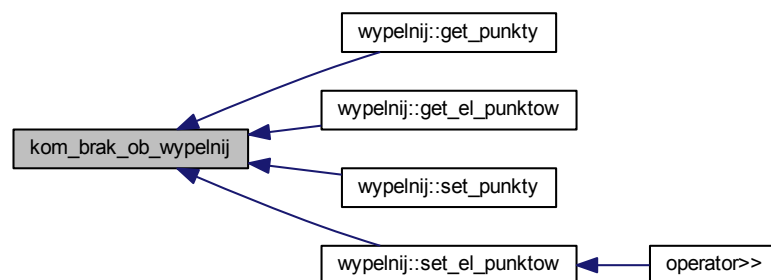
Funkcja zwraca napis informujący o wyjątku w klasie `element_planszy`. Napis informuje o braku dostępu do obiektu klasy `element_planszy`. Oto graf wywoływań tej funkcji:



#### 5.11.1.4 kom\_brak\_ob\_wypelnij()

```
std::string kom_brak_ob_wypelnij ( )
```

Funkcja zwraca napis informujący o wyjątku w klasie `wypelnij`. Napis informuje o braku dostępu do obiektu klasy `wypelnij`. Oto graf wywołań tej funkcji:



#### 5.11.1.5 kom\_brak\_ob\_wyswietl()

```
std::string kom_brak_ob_wyswietl ( )
```

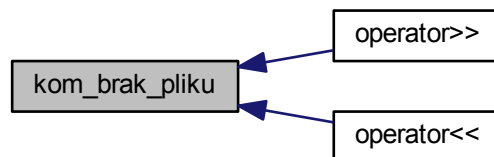
Funkcja zwraca napis informujący o wyjątku w klasie `wyswietl`. Napis informuje o braku dostępu do obiektu klasy `wyswietl`. Oto graf wywołań tej funkcji:



#### 5.11.1.6 `kom_brak_pliku()`

```
std::string kom_brak_pliku ( )
```

Funkcja zwraca napis informujący o wyjątku w przeciążonym operatorze. Napis informuje o braku dostępu do pliku o podanej nazwie. Oto graf wywołań tej funkcji:



#### 5.11.1.7 `kom_brak_wsp_y()`

```
std::string kom_brak_wsp_y ( )
```

Funkcja zwraca napis informujący o wyjątku w klasie `wypelnij`. Napis informuje o braku drugiej współrzędnej komórki do ożywienia. Oto graf wywołań tej funkcji:



#### 5.11.1.8 kom\_wsp\_spoza\_planszy()

```
std::string kom_wsp_spoza_planszy ( )
```

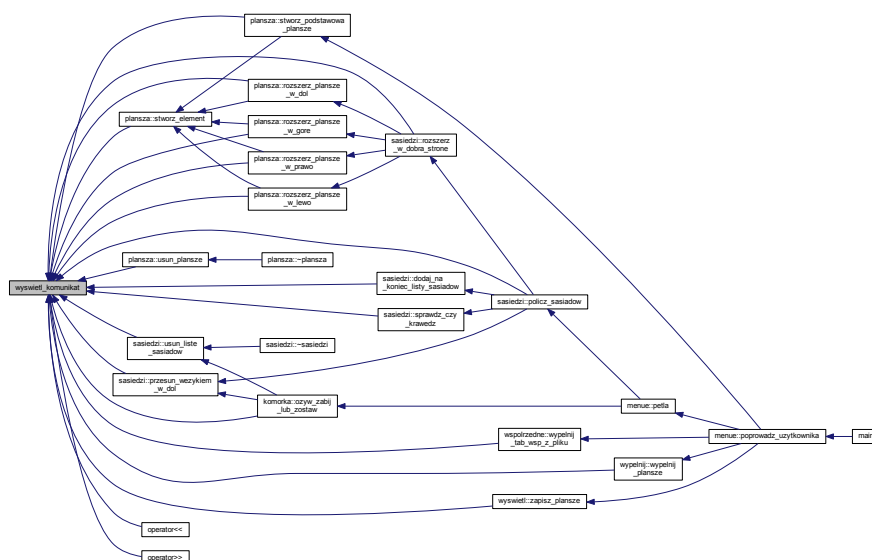
Funkcja zwraca napis informujący o wyjątku w klasie wypelnij. Napis informuje o tym, że współrzędna komórki do ożywienia wskazuje poza plansze. Wyjątek jest wyrzucony w metodzie wczytującej dane z pliku(przy ręcznym wpisywaniu jest to od razu sprawdzane). Oto graf wywoływań tej funkcji:



#### 5.11.1.9 wyswietl\_komunikat()

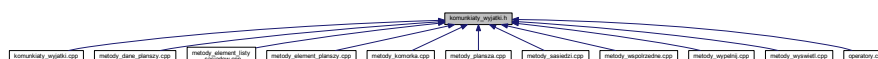
```
void wyswietl_komunikat (
    std::string str )
```

Funkcja wyświetla stringa o podanej nazwie. Jest używana do wyświetlania ewentualnych wyjątków. Wyświetla inny komunikat w zależności od tego, jaki wyjątek został wyrzucony. Oto graf wywołań tej funkcji:



## 5.12 Dokumentacja pliku komunikaty\_wyjatki.h

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



## Funkcje

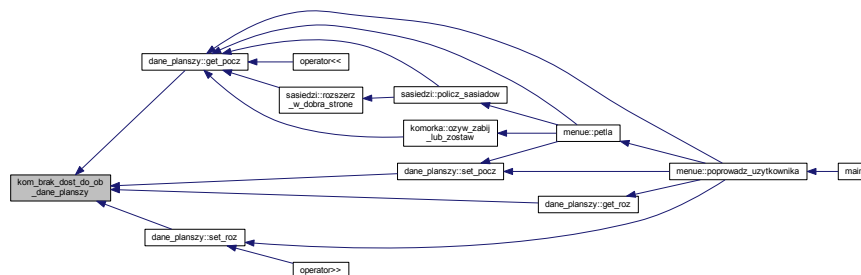
- void `wyswietl_komunikat` (std::string str)
- std::string `kom_brak_dost_do_ob_dane_planszy` ()
- std::string `kom_brak_ob_element_planszy` ()
- std::string `kom_brak_ob_wypelnij` ()
- std::string `kom_wsp_spoza_planszy` ()
- std::string `kom_brak_wsp_y` ()
- std::string `kom_brak_ob_wyswietl` ()
- std::string `kom_brak_pliku` ()
- std::string `kom_brak_ob_element_listy_sasiadow` ()

### 5.12.1 Dokumentacja funkcji

#### 5.12.1.1 `kom_brak_dost_do_ob_dane_planszy()`

```
std::string kom_brak_dost_do_ob_dane_planszy ( )
```

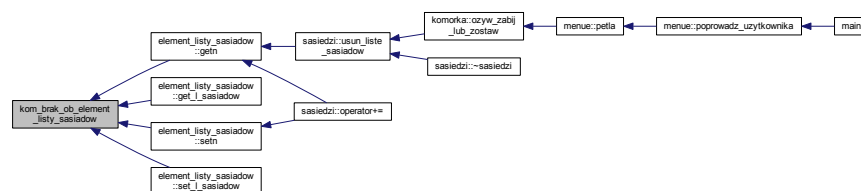
Funkcja zwraca napis informujący o wyjątku w klasie `dane_planszy`. Napis informuje o braku dostępu do obiektu klasy `dane_planszy`. Oto graf wywołań tej funkcji:



#### 5.12.1.2 `kom_brak_ob_element_listy_sasiadow()`

```
std::string kom_brak_ob_element_listy_sasiadow ( )
```

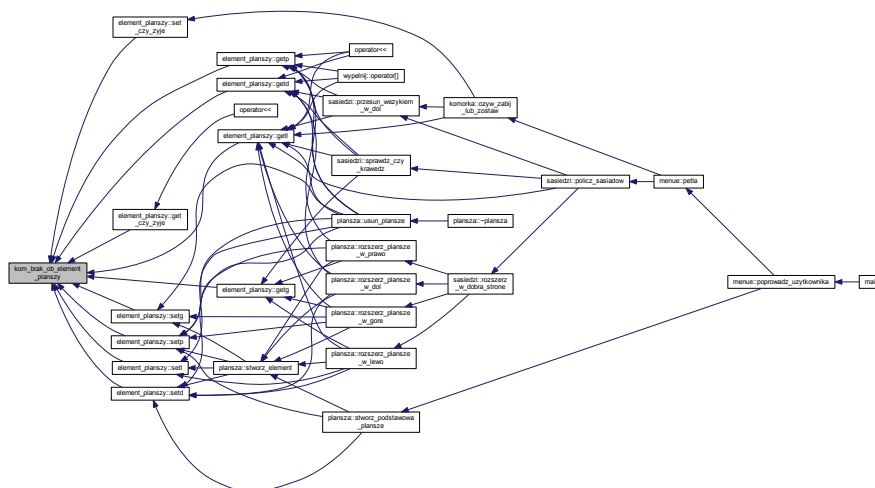
Funkcja zwraca napis informujący o wyjątku w klasie `element_listy_sasiadow`. Napis informuje o braku dostępu do obiektu klasy `element_listy_sasiadow`. Oto graf wywołań tej funkcji:



### 5.12.1.3 kom\_brak\_ob\_element\_planszy()

```
std::string kom_brak_ob_element_planszy ( )
```

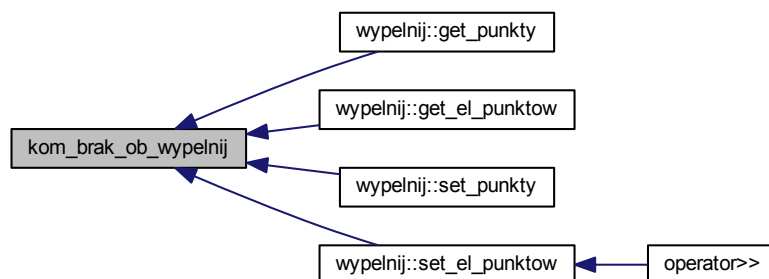
Funkcja zwraca napis informujący o wyjątku w klasie `element_planszy`. Napis informuje o braku dostępu do obiektu klasy `element_planszy`. Oto graf wywoływań tej funkcji:



#### 5.12.1.4 kom\_brak\_ob\_wypelnij()

```
std::string kom_brak_ob_wypelnij ( )
```

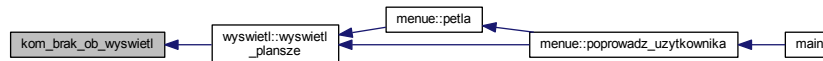
Funkcja zwraca napis informujący o wyjątku w klasie `wypelnij`. Napis informuje o braku dostępu do obiektu klasy `wypelnij`. Oto graf wywołań tej funkcji:



#### 5.12.1.5 kom\_brak\_ob\_wyswietl()

```
std::string kom_brak_ob_wyswietl ( )
```

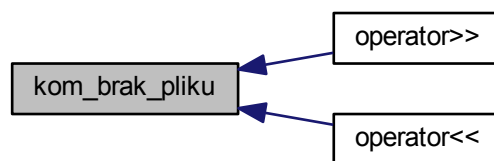
Funkcja zwraca napis informujący o wyjątku w klasie wyswietl. Napis informuje o braku dostępu do obiektu klasy wyswietl. Oto graf wywołań tej funkcji:



#### 5.12.1.6 kom\_brak\_pliku()

```
std::string kom_brak_pliku ( )
```

Funkcja zwraca napis informujący o wyjątku w przeciążonym operatorze. Napis informuje o braku dostępu do pliku o podanej nazwie. Oto graf wywołań tej funkcji:



#### 5.12.1.7 kom\_brak\_wsp\_y()

```
std::string kom_brak_wsp_y ( )
```

Funkcja zwraca napis informujący o wyjątku w klasie wypelnij. Napis informuje o braku drugiej współrzędnej komórki do ozywienia. Oto graf wywołań tej funkcji:





#### 5.12.1.8 kom\_wsp\_spoza\_planszy()

```
std::string kom_wsp_spoza_planszy ( )
```

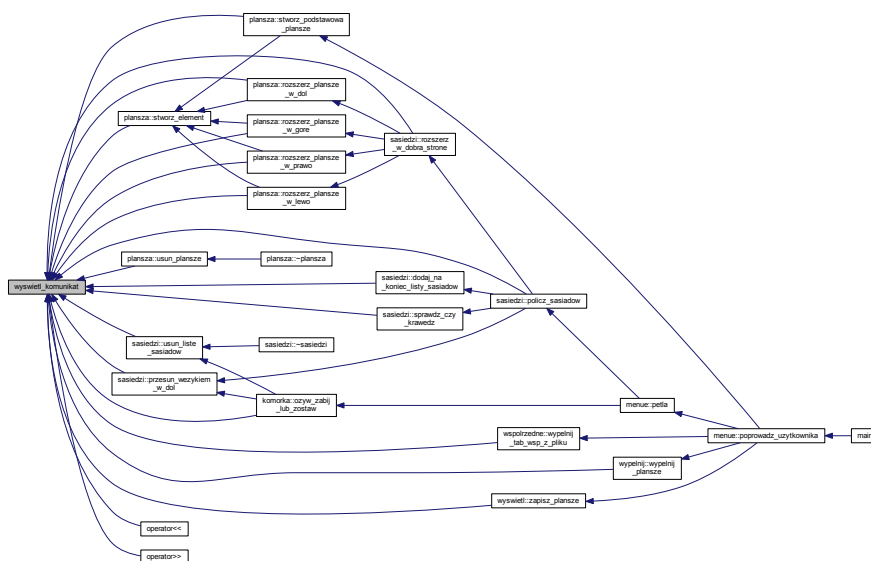
Funkcja zwraca napis informujący o wyjątku w klasie wypelnij. Napis informuje o tym, że współrzędna komórki do ożywienia wskazuje poza plansze. Wyjątek jest wyrzucony w metodzie wczytującej dane z pliku(przy ręcznym wpisywaniu jest to od razu sprawdzane). Oto graf wywoływań tej funkcji:



#### 5.12.1.9 wyswietl\_komunikat()

```
void wyswietl_komunikat (
    std::string str )
```

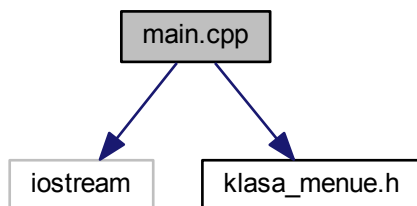
Funkcja wyświetla stringa o podanej nazwie. Jest używana do wyświetlania ewentualnych wyjątków. Wyświetla inny komunikat w zależności od tego, jaki wyjątek został wyrzucony. Oto graf wywołań tej funkcji:



## 5.13 Dokumentacja pliku main.cpp

```
#include <iostream>
#include "klasa_menu.h"
```

Wykres zależności załączania dla main.cpp:



### Funkcje

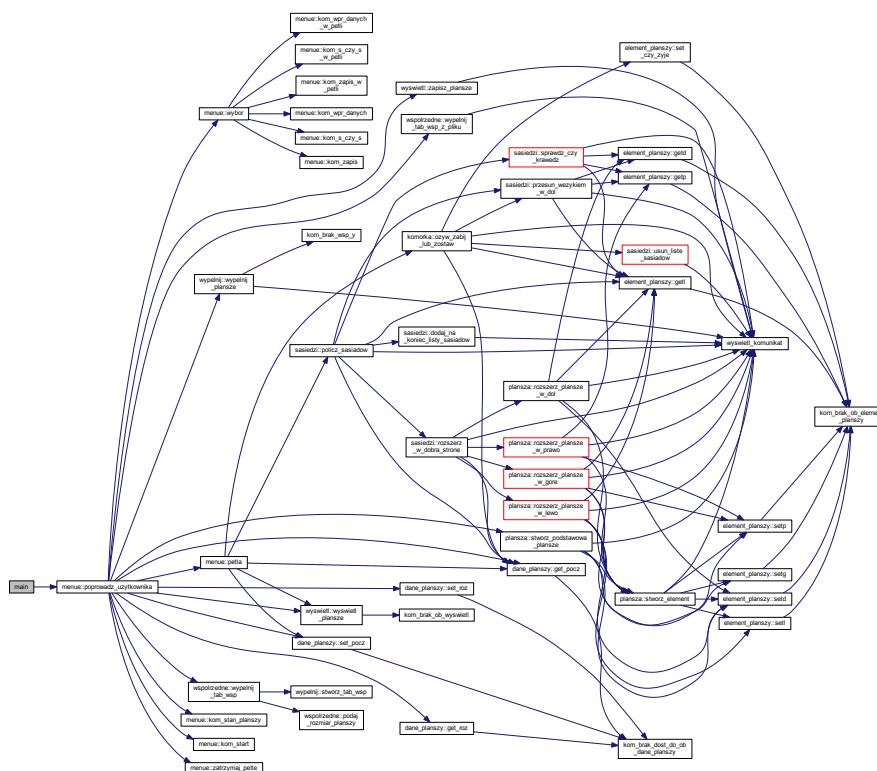
- int `main` ()

#### 5.13.1 Dokumentacja funkcji

##### 5.13.1.1 main()

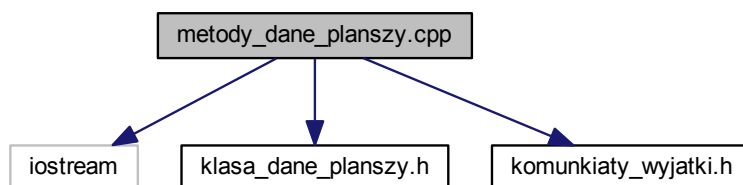
```
int main ( )
```

Oto graf wywołań dla tej funkcji:



## 5.14 Dokumentacja pliku metody\_dane\_planszy.cpp

```
#include <iostream>
#include "klasa_dane_planszy.h"
#include "komunkiaty_wyjatki.h"
Wykres zależności załączania dla metody dane_planszy.cpp:
```

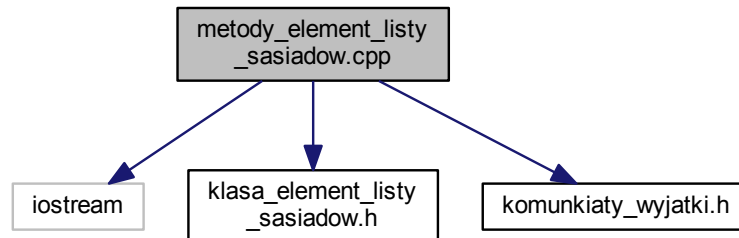


### 5.15 Dokumentacja pliku metody\_element\_listy\_sasiadow.cpp

```
#include <iostream>
#include "klasa_element_listy_sasiadow.h"
```

```
#include "komunkiaty_wyjatki.h"
```

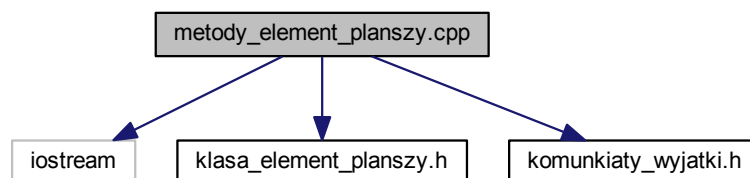
Wykres zależności załączania dla metody\_element\_listy\_sasiadow.cpp:



## 5.16 Dokumentacja pliku metody\_element\_planszy.cpp

```
#include <iostream>
#include "klasa_element_planszy.h"
#include "komunkiaty_wyjatki.h"
```

Wykres zależności załączania dla metody\_element\_planszy.cpp:

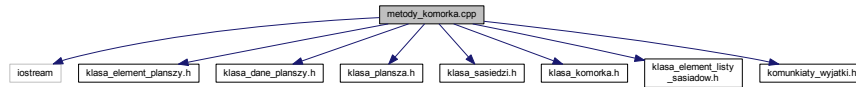


## 5.17 Dokumentacja pliku metody\_komorka.cpp

```
#include <iostream>
#include "klasa_element_planszy.h"
#include "klasa_dane_planszy.h"
#include "klasa_plansza.h"
#include "klasa_sasiedzi.h"
#include "klasa_komorka.h"
#include "klasa_element_listy_sasiadow.h"
```

```
#include "komunkiaty_wyjatki.h"
```

Wykres zależności załączania dla metody\_komorka.cpp:



## 5.18 Dokumentacja pliku metody\_menue.cpp

```
#include <iostream>
#include "klasa_menue.h"
#include <thread>
#include <functional>
#include "klasa_element_planszy.h"
#include "klasa_dane_planszy.h"
#include "klasa_plansza.h"
#include "klasa_wypelnij.h"
#include "klasa_wspolrzedne.h"
#include "klasa_wyswietl.h"
#include "klasa_sasiedzi.h"
#include "klasa_komorka.h"
#include "operator.h"
#include <stdlib.h>
#include "windows.h"
```

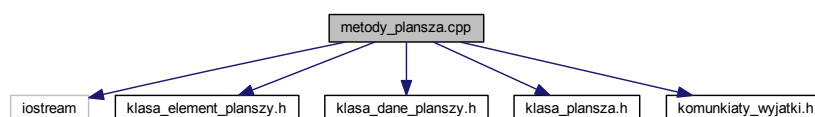
Wykres zależności załączania dla metody\_menue.cpp:



## 5.19 Dokumentacja pliku metody\_plansza.cpp

```
#include <iostream>
#include "klasa_element_planszy.h"
#include "klasa_dane_planszy.h"
#include "klasa_plansza.h"
#include "komunkiaty_wyjatki.h"
```

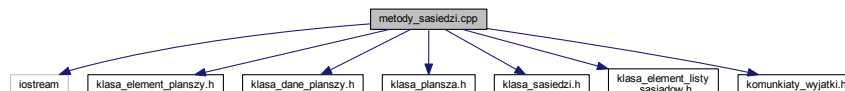
Wykres zależności załączania dla metody\_plansza.cpp:



## 5.20 Dokumentacja pliku metody\_sasiedzi.cpp

```
#include <iostream>
#include "klasa_element_planszy.h"
#include "klasa_dane_planszy.h"
#include "klasa_plansza.h"
#include "klasa_sasiedzi.h"
#include "klasa_element_listy_sasiadow.h"
#include "komunkiaty_wyjatki.h"
```

Wykres zależności załączania dla metody\_sasiedzi.cpp:



## 5.21 Dokumentacja pliku metody\_wspolrzedne.cpp

```
#include <iostream>
#include "klasa_dane_planszy.h"
#include "klasa_wypelnij.h"
#include "klasa_wspolrzedne.h"
#include <string>
#include <fstream>
#include "operator.h"
#include "komunkiaty_wyjatki.h"
```

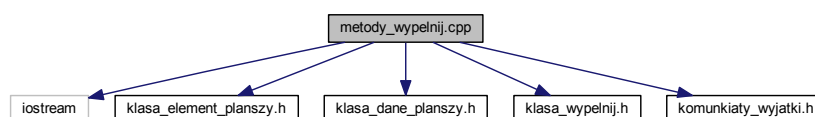
Wykres zależności załączania dla metody\_wspolrzedne.cpp:



## 5.22 Dokumentacja pliku metody\_wypelnij.cpp

```
#include <iostream>
#include "klasa_element_planszy.h"
#include "klasa_dane_planszy.h"
#include "klasa_wypelnij.h"
#include "komunkiaty_wyjatki.h"
```

Wykres zależności załączania dla metody\_wypelnij.cpp:



## 5.23 Dokumentacja pliku metody\_wyswietl.cpp

```
#include <iostream>
#include "klasa_element_planszy.h"
#include "klasa_dane_planszy.h"
#include "klasa_wyswietl.h"
#include "operator.h"
#include <string>
#include <fstream>
#include "komunkiaty_wyjatki.h"
```

Wykres zależności załączania dla metody\_wyswietl.cpp:



## 5.24 Dokumentacja pliku operator.cpp

```
#include <iostream>
#include "operator.h"
#include "klasa_element_planszy.h"
#include "klasa_dane_planszy.h"
#include "klasa_wyswietl.h"
#include "klasa_wypelnij.h"
#include "klasa_wspolrzedne.h"
#include <string>
#include <fstream>
#include "komunkiaty_wyjatki.h"
```

Wykres zależności załączania dla operator.cpp:



### Funkcje

- std::ostream & [operator<<](#) (std::ostream &s, [element\\_planszy](#) \*ob)
- std::ostream & [operator<<](#) (std::ostream &s, [wyswietl](#) &ob)
- std::ifstream & [operator>>](#) (std::ifstream &plik, [wspolrzedne](#) &ob)
- std::ofstream & [operator<<](#) (std::ofstream &plik, [wyswietl](#) &ob)

### 5.24.1 Dokumentacja funkcji

### 5.24.1.1 operator<<() [1/3]

```
std::ostream& operator<< (
    std::ostream & s,
    element_planszy * ob )
```

Przeciążony operator strumieniowy << służy do wyświetlania stanu obiektu klasy `element_planszy`.

W zależności od stanu (wartości pola `czy_zyje`) elementu wskazywanego przez drugi argument operator "wrzuca" do strumienia ostream inne wartości. "." - gdy komórka nie żyje. "O" - gdy komórka żyje.

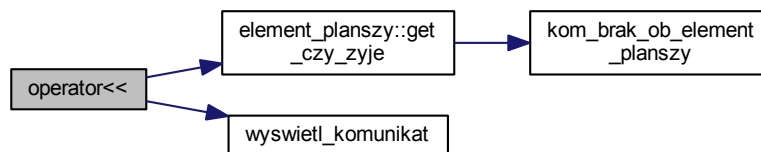
#### Parametry

<code>s</code>	strumień ostream, do którego będą "wrzucone" znaki do wyświetlenia.
<code>ob</code>	wskaznik na element, którego stan ma być wyświetlony.

#### Zwraca

strumień ostream i to co w nim.

Oto graf wywołań dla tej funkcji:



### 5.24.1.2 operator<<() [2/3]

```
std::ostream& operator<< (
    std::ostream & s,
    wyswietl & ob )
```

Przeciążony operator strumieniowy << służy do wyświetlania aktualnego stanu planszy do gry w życie.

Operator ustawia wskaznik pomocniczy na pierwszy element planszy (w obiekcie wyswietl). Przesuwa go po niej i używa przeciążonego operatora << dla elementu planszy do wyświetlenia jego stanu.

#### Parametry

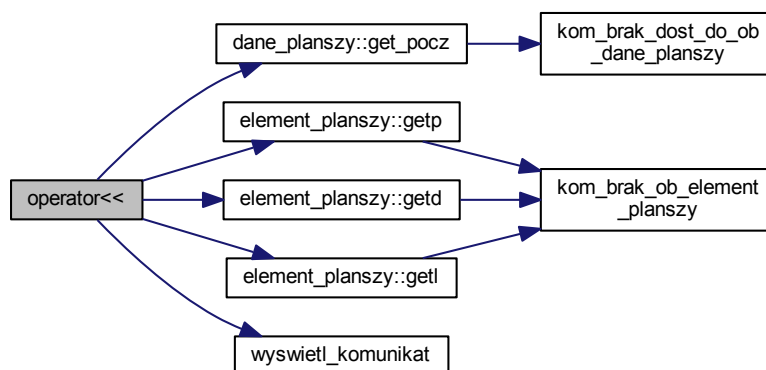
<code>s</code>	strumień ostream, do którego będą "wrzucone" znaki do wyświetlenia.
<code>ob</code>	obiekt klasy wypelnij którego stan planszy pod wskaznikiem ppoczątkowy być wyświetlony.



**Zwraca**

strumień `ostream` i to co w nim.

Oto graf wywołań dla tej funkcji:

**5.24.1.3 operator<<()** [3/3]

```
std::ostream& operator<< (
    std::ostream & plik,
    wyswietl & ob )
```

Przeciążony operator strumieniowy `<<` służy do zapisania aktualnego stanu planszy do pliku.

Jesli plik podany jako pierwszy argument istnieje oraz wskaźnik początkowy obiektu podanego jako drugi argument na coś wskazuje, operator zapisuje w pliku aktualny stan planszy wskazanej przez wskaźnik początkowy obiektu klasy współrzędne. Operator używa pomocniczego wskaźnika którym przesuwa się po planszy i używa przeciążonego operatora `<<` dla każdego elementu planszy. Jesli nie ma planszy, operator nic nie robi. Jesli nie istnieje podany plik, operator wyrzuca wyjątek.

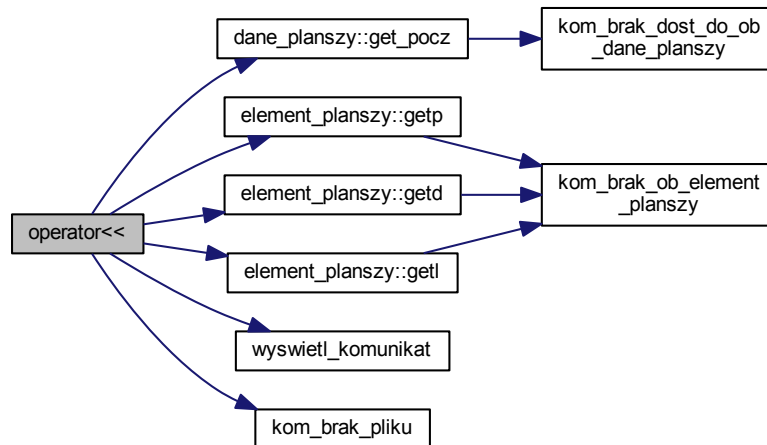
**Parametry**

<i>plik</i>	strumień <code>ofstream</code> , plik do którego zapisany będzie aktualny stan planszy.
<i>ob</i>	obiekt klasy <code>wyswietl</code> , z którego ma zostać zapisany stan planszy (wskazywanej przez wskaźnik początkowy).

**Zwraca**

strumien ofstream.

Oto graf wywołań dla tej funkcji:

**5.24.1.4 operator>>()**

```

std::ifstream& operator>> (
    std::ifstream & plik,
    wspolrzedne & ob )

```

Przeciazony operator strumieniowy >> sluzzy do wczytywania rozmiaru kwadratowej planszy poczatkowej oraz wspolrzecznych komorek do ozywienia z pliku.

Jesli plik podany jako pierwszy argument istnieje, operator sczytuje rozmiar kwadratowej planszy podstawowej. Nastepnie tworzy tablice z wspolrzednymi(pole punkty(wywołanie metody stworz\_tab\_wsp dla ob klasy wspolrzedne)). Natepnie dopoki w pliku istnieja jakies liczby operator sczytuje je do stworzonej tablicy. Operator wyrzuca wyjatek gdy nie ma dostepu do pliku podanego jako pierwszy argument.

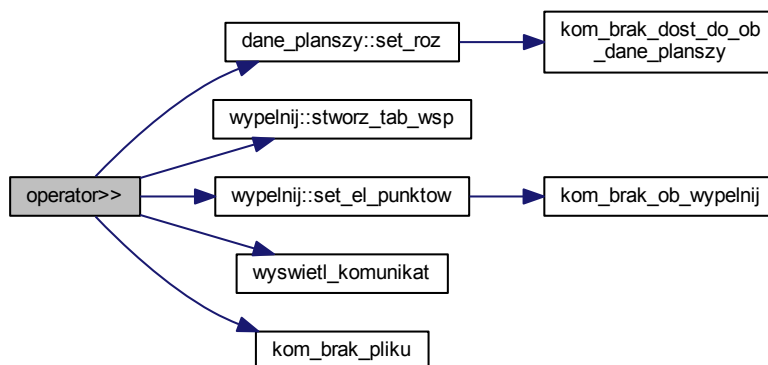
**Parametry**

<i>plik</i>	strumien ifstream, plik z ktorego beda wczytane dane.
<i>ob</i>	obiekt klasy wspolrzedne, do ktorego (do tablicy punkty) maja zostac wczytane wspolrzedne.

Zwraca

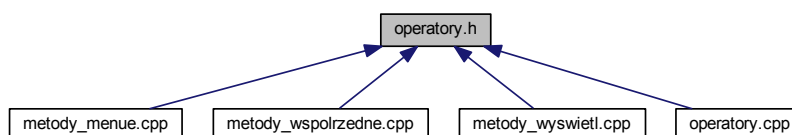
strumień ifstream.

Oto graf wywołań dla tej funkcji:



## 5.25 Dokumentacja pliku operator.h

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



### Funkcje

- std::ostream & [operator<<](#) (std::ostream &s, [element\\_planszy](#) \*ob)
- std::ostream & [operator<<](#) (std::ostream &s, [wyswietl](#) &ob)
- std::ifstream & [operator>>](#) (std::ifstream &plik, [wspolrzedne](#) &ob)
- std::ofstream & [operator<<](#) (std::ofstream &plik, [wyswietl](#) &ob)

### 5.25.1 Dokumentacja funkcji

### 5.25.1.1 operator<<() [1/3]

```
std::ostream& operator<< (
    std::ostream & s,
    element_planszy * ob )
```

Przeciążony operator strumieniowy << służy do wyświetlania stanu obiektu klasy `element_planszy`.

W zależności od stanu (wartości pola `czy_zyje`) elementu wskazywanego przez drugi argument operator "wrzuca" do strumienia ostream inne wartości. "." - gdy komórka nie żyje. "O" - gdy komórka żyje.

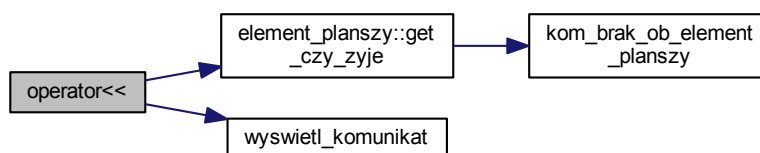
#### Parametry

<code>s</code>	strumień ostream, do którego będą "wrzucone" znaki do wyświetlenia.
<code>ob</code>	wskaznik na element, którego stan ma być wyświetlony.

#### Zwraca

strumień ostream i to co w nim.

Oto graf wywołań dla tej funkcji:



### 5.25.1.2 operator<<() [2/3]

```
std::ostream& operator<< (
    std::ostream & s,
    wyswietl & ob )
```

Przeciążony operator strumieniowy << służy do wyświetlania aktualnego stanu planszy do gry w życie.

Operator ustawia wskaznik pomocniczy na pierwszy element planszy (w obiekcie wyswietl). Przesuwa go po niej i używa przeciążonego operatora << dla elementu planszy do wyświetlenia jego stanu.

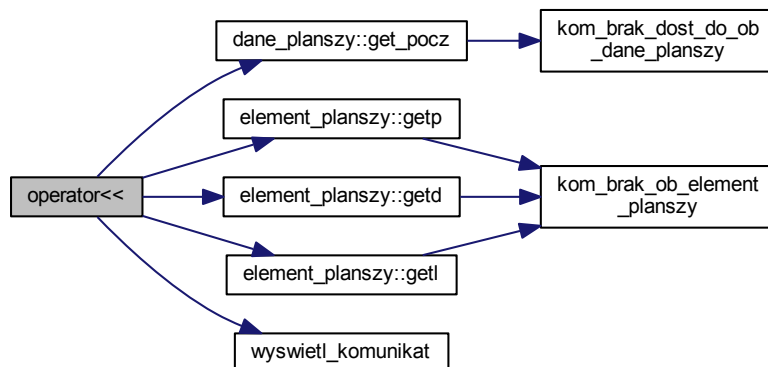
#### Parametry

<code>s</code>	strumień ostream, do którego będą "wrzucone" znaki do wyświetlenia.
<code>ob</code>	obiekt klasy wypelnij którego stan planszy pod wskaznikiem ppoczątkowy być wyświetlony.

**Zwraca**

strumien ostream i to co w nim.

Oto graf wywołań dla tej funkcji:

**5.25.1.3 operator<<() [3/3]**

```
std::ostream& operator<< (
    std::ostream & plik,
    wyswietl & ob )
```

Przeciazony operator strumieniowy << sluzi do zapisania aktualnego stanu planszy do pliku.

Jesli plik podany jako pierwszy argument istnieje oraz wskaznik ppocatkowy obiektu podanego jako drugi argument na cos wskazuje, operator zapisuje w pliku aktualny stan planszy wskazanej przez wskaznik ppocatkowy obiektu klasy wspolrzedne. Operator uzywa pomocniczego wskaznika ktorym przesuwaja sie po planszy i uzywa przeciazonego operatora << dla kazdego elementu planszy. Jesli nie ma planszy, operator nic nie robi. Jesli nie istnieje podany plik, operator wyrzuca wyjatek.

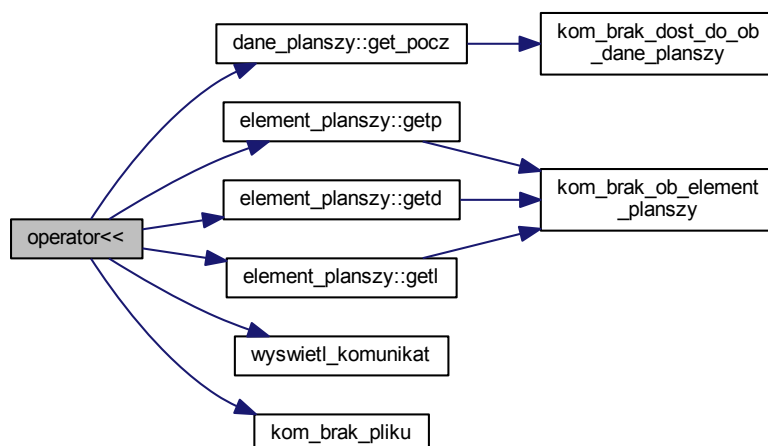
**Parametry**

<i>plik</i>	strumien ostream, plik do ktorego zapisany bedzie aktualny stan planszy.
<i>ob</i>	obiekt klasy wyswietl, z ktorego ma zostac zapisany stan planszy(wskazywanej przez wskaznik ppocatkowy).

**Zwraca**

strumien ofstream.

Oto graf wywołań dla tej funkcji:

**5.25.1.4 operator>>()**

```

std::ifstream& operator>> (
    std::ifstream & plik,
    wspolrzedne & ob )

```

Przeciazony operator strumieniowy >> sluzzy do wczytywania rozmiaru kwadratowej planszy poczatkowej oraz wspolrzecznych komorek do ozywienia z pliku.

Jesli plik podany jako pierwszy argument istnieje, operator sczytuje rozmiar kwadratowej planszy podstawowej. Nastepnie tworzy tablice z wspolrzednymi(pole punkty(wywołanie metody stworz\_tab\_wsp dla ob klasy wspolrzedne)). Natepnie dopoki w pliku istnieja jakies liczby operator sczytuje je do stworzonej tablicy. Operator wyrzuca wyjatek gdy nie ma dostepu do pliku podanego jako pierwszy argument.

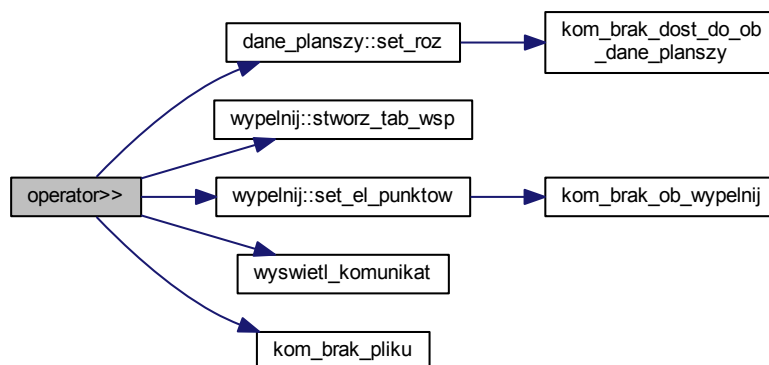
**Parametry**

<i>plik</i>	strumien ifstream, plik z ktorego beda wczytane dane.
<i>ob</i>	obiekt klasy wspolrzedne, do ktorego (do tablicy punkty) maja zostac wczytane wspolrzedne.

Zwraca

strumień ifstream.

Oto graf wywołań dla tej funkcji:







# Skorowidz

~plansza  
  plansza, [41](#)  
~sasiedzi  
  sasiedzi, [51](#)  
~wypelnij  
  wypelnij, [66](#)

dane\_planszy, [7](#)  
  get\_pocz, [9](#)  
  get\_roz, [9](#)  
  ppoczatkowy, [12](#)  
  rozmiar, [12](#)  
  set\_pocz, [10](#)  
  set\_roz, [11](#)

dodaj\_na\_koniec\_listy\_sasiadow  
  sasiedzi, [51](#)

element\_listy\_sasiadow, [13](#)  
  element\_listy\_sasiadow, [13](#)  
  get\_l\_sasiadow, [14](#)  
  getn, [14](#)  
  set\_l\_sasiadow, [15](#)  
  setn, [16](#)

element\_planszy, [17](#)  
  get\_czy\_zyje, [18](#)  
  getd, [18](#)  
  getg, [19](#)  
  getl, [20](#)  
  getp, [21](#)  
  set\_czy\_zyje, [21](#)  
  setd, [22](#)  
  setg, [23](#)  
  setl, [24](#)  
  setp, [25](#)

get\_czy\_zyje  
  element\_planszy, [18](#)

get\_el\_punktow  
  wypelnij, [66](#)

get\_l\_sasiadow  
  element\_listy\_sasiadow, [14](#)

get\_pocz  
  dane\_planszy, [9](#)

get\_punkty  
  wypelnij, [67](#)

get\_roz  
  dane\_planszy, [9](#)

getd  
  element\_planszy, [18](#)

getg  
  element\_planszy, [19](#)

getl  
  element\_planszy, [20](#)

getn  
  element\_listy\_sasiadow, [14](#)

getp  
  element\_planszy, [21](#)

klasa\_dane\_planszy.h, [77](#)  
klasa\_element\_listy\_sasiadow.h, [77](#)  
klasa\_element\_planszy.h, [78](#)  
klasa\_komorka.h, [78](#)  
klasa\_menu.h, [78](#)  
klasa\_plansza.h, [79](#)  
klasa\_sasiedzi.h, [79](#)  
klasa\_wspolrzedne.h, [80](#)  
klasa\_wypelnij.h, [80](#)  
klasa\_wyswietl.h, [80](#)  
kom\_brak\_dost\_do\_ob\_dane\_planszy  
  komunkiaty\_wyjatki.cpp, [81](#)  
  komunkiaty\_wyjatki.h, [86](#)  
kom\_brak\_ob\_element\_listy\_sasiadow  
  komunkiaty\_wyjatki.cpp, [82](#)  
  komunkiaty\_wyjatki.h, [86](#)  
kom\_brak\_ob\_element\_planszy  
  komunkiaty\_wyjatki.cpp, [82](#)  
  komunkiaty\_wyjatki.h, [86](#)  
kom\_brak\_ob\_wypelnij  
  komunkiaty\_wyjatki.cpp, [83](#)  
  komunkiaty\_wyjatki.h, [87](#)  
kom\_brak\_ob\_wyswietl  
  komunkiaty\_wyjatki.cpp, [83](#)  
  komunkiaty\_wyjatki.h, [87](#)  
kom\_brak\_pliku  
  komunkiaty\_wyjatki.cpp, [84](#)  
  komunkiaty\_wyjatki.h, [88](#)  
kom\_brak\_wsp\_y  
  komunkiaty\_wyjatki.cpp, [84](#)  
  komunkiaty\_wyjatki.h, [88](#)  
kom\_s\_czy\_s  
  menu, [30](#)  
kom\_s\_czy\_s\_w\_petli  
  menu, [31](#)  
kom\_stan\_planszy  
  menu, [31](#)  
kom\_start  
  menu, [32](#)  
kom\_wpr\_danych  
  menu, [32](#)  
kom\_wpr\_danych\_w\_petli

- menue, 33
- kom\_wsp\_spoza\_planszy
  - komunkiaty\_wyjatki.cpp, 84
  - komunkiaty\_wyjatki.h, 88
- kom\_zapis
  - menue, 33
- kom\_zapis\_w\_petli
  - menue, 34
- komorka, 26
  - ozyw\_zabij\_lub\_zostaw, 29
- komunkiaty\_wyjatki.cpp, 81
  - kom\_brak\_dost\_do\_ob\_dane\_planszy, 81
  - kom\_brak\_ob\_element\_listy\_sasiadow, 82
  - kom\_brak\_ob\_element\_planszy, 82
  - kom\_brak\_ob\_wypelnij, 83
  - kom\_brak\_ob\_wyswietl, 83
  - kom\_brak\_pliku, 84
  - kom\_brak\_wsp\_y, 84
  - kom\_wsp\_spoza\_planszy, 84
  - wyswietl\_komunikat, 85
- komunkiaty\_wyjatki.h, 85
  - kom\_brak\_dost\_do\_ob\_dane\_planszy, 86
  - kom\_brak\_ob\_element\_listy\_sasiadow, 86
  - kom\_brak\_ob\_element\_planszy, 86
  - kom\_brak\_ob\_wypelnij, 87
  - kom\_brak\_ob\_wyswietl, 87
  - kom\_brak\_pliku, 88
  - kom\_brak\_wsp\_y, 88
  - kom\_wsp\_spoza\_planszy, 88
  - wyswietl\_komunikat, 89
- main
  - main.cpp, 90
- main.cpp, 90
  - main, 90
- menue, 30
  - kom\_s\_czy\_s, 30
  - kom\_s\_czy\_s\_w\_petli, 31
  - kom\_stan\_planszy, 31
  - kom\_start, 32
  - kom\_wpr\_danych, 32
  - kom\_wpr\_danych\_w\_petli, 33
  - kom\_zapis, 33
  - kom\_zapis\_w\_petli, 34
  - petla, 34
  - poprowadz\_uzytkownika, 35
  - wybor, 37
  - zatrzymaj\_petle, 38
- metody\_dane\_planszy.cpp, 91
- metody\_element\_listy\_sasiadow.cpp, 91
- metody\_element\_planszy.cpp, 92
- metody\_komorka.cpp, 92
- metody\_menue.cpp, 93
- metody\_plansza.cpp, 93
- metody\_sasiedzi.cpp, 94
- metody\_wspolrzedne.cpp, 94
- metody\_wypelnij.cpp, 94
- metody\_wyswietl.cpp, 95
- ob
  - sasiedzi, 58
- operator int
  - plansza, 41
- operator<<
  - operator.cpp, 95–97
  - operator.h, 99–101
- operator>>
  - operator.cpp, 98
  - operator.h, 102
- operator+=
  - sasiedzi, 52
- operator[]
  - wypelnij, 67
- operator.cpp, 95
  - operator<<, 95–97
  - operator>>, 98
- operator.h, 99
  - operator<<, 99–101
  - operator>>, 102
- ozyw\_zabij\_lub\_zostaw
  - komorka, 29
- petla
  - menue, 34
- pglowa
  - sasiedzi, 58
- plansza, 39
  - ~plansza, 41
  - operator int, 41
  - rozszerz\_plansze\_w\_dol, 42
  - rozszerz\_plansze\_w\_gore, 43
  - rozszerz\_plansze\_w\_lewo, 44
  - rozszerz\_plansze\_w\_prawo, 45
  - stworz\_element, 46
  - stworz\_podstawowa\_plansze, 47
  - usun\_plansze, 47
- podaj\_rozmiar\_planszy
  - wspolrzedne, 61
- policz\_sasiadow
  - sasiedzi, 53
- poprowadz\_uzytkownika
  - menue, 35
- ppoczatkowy
  - dane\_planszy, 12
- przesun\_wezykiem\_w\_dol
  - sasiedzi, 54
- punkty
  - wypelnij, 72
- rozmiar
  - dane\_planszy, 12
- rozszerz\_plansze\_w\_dol
  - plansza, 42
- rozszerz\_plansze\_w\_gore
  - plansza, 43
- rozszerz\_plansze\_w\_lewo
  - plansza, 44
- rozszerz\_plansze\_w\_prawo

plansza, [45](#)  
rozszerz\_w\_dobra\_strone  
  sasiedzi, [55](#)  
sasiedzi, [49](#)  
  ~sasiedzi, [51](#)  
  dodaj\_na\_koniec\_listy\_sasiadow, [51](#)  
  ob, [58](#)  
  operator+=", [52](#)  
  pglowa, [58](#)  
  policz\_sasiadow, [53](#)  
  przesun\_wezykiem\_w\_dol, [54](#)  
  rozszerz\_w\_dobra\_strone, [55](#)  
  sasiedzi, [51](#)  
  sprawdz\_czy\_krawedz, [56](#)  
  usun\_liste\_sasiadow, [57](#)  
set\_czy\_zyje  
  element\_planszy, [21](#)  
set\_el\_punktow  
  wypelnij, [68](#)  
set\_l\_sasiadow  
  element\_listy\_sasiadow, [15](#)  
set\_pocz  
  dane\_planszy, [10](#)  
set\_punkty  
  wypelnij, [69](#)  
set\_roz  
  dane\_planszy, [11](#)  
setd  
  element\_planszy, [22](#)  
setg  
  element\_planszy, [23](#)  
setl  
  element\_planszy, [24](#)  
setn  
  element\_listy\_sasiadow, [16](#)  
setp  
  element\_planszy, [25](#)  
sprawdz\_czy\_krawedz  
  sasiedzi, [56](#)  
stworz\_element  
  plansza, [46](#)  
stworz\_podstawowa\_plansze  
  plansza, [47](#)  
stworz\_tab\_wsp  
  wypelnij, [70](#)  
  
usun\_liste\_sasiadow  
  sasiedzi, [57](#)  
usun\_plansze  
  plansza, [47](#)  
usun\_tab\_wsp  
  wypelnij, [70](#)  
  
wspolzedne, [59](#)  
  podaj\_rozmiar\_planszy, [61](#)  
  wypelnij\_tab\_wsp, [61](#)  
  wypelnij\_tab\_wsp\_z\_pliku, [62](#)  
wybor  
  
  menue, [37](#)  
wypelnij, [63](#)  
  ~wypelnij, [66](#)  
  get\_el\_punktow, [66](#)  
  get\_punkty, [67](#)  
  operator[], [67](#)  
  punkty, [72](#)  
  set\_el\_punktow, [68](#)  
  set\_punkty, [69](#)  
  stworz\_tab\_wsp, [70](#)  
  usun\_tab\_wsp, [70](#)  
  wypelnij, [66](#)  
  wypelnij\_plansze, [71](#)  
wypelnij\_plansze  
  wypelnij, [71](#)  
wypelnij\_tab\_wsp  
  wspolzedne, [61](#)  
wypelnij\_tab\_wsp\_z\_pliku  
  wspolzedne, [62](#)  
wyswietl, [72](#)  
  wyswietl\_plansze, [74](#)  
  zapisz\_plansze, [74](#)  
wyswietl\_komunikat  
  komunkiaty\_wyjatki.cpp, [85](#)  
  komunkiaty\_wyjatki.h, [89](#)  
wyswietl\_plansze  
  wyswietl, [74](#)  
  
zapisz\_plansze  
  wyswietl, [74](#)  
zatrzymaj\_petle  
  menue, [38](#)