Link do programu:

https://www.dropbox.com/sh/f7x0ycjy9dbbu2n/AADf00kzQBT_rpggn6PWaVgta?dl=0

Politechnika Śląska
Wydział Automatyki, Elektroniki i Informatyki

Programowanie Komputerów 3

Sudoku

autor	Hanna Podeszwa
prowadzący	mgr inż. Grzegorz Kwiatkowski
rok akademicki	2019/2020
kierunek	informatyka
rodzaj studiów	SSI
semestr	3
termin laboratorium	wtorek, 13:45 – 15:15
termin oddania sprawozdania	2020-01-08

1 Treść zadania

Napisać program pozwalający rozwiązywać sudoku. Gracz może to zrobić samodzielnie lub skorzystać z rozwiązywania sudoku przez program. Możliwe jest też utworzenie nowej planszy do gry.

2 Analiza zadania

Zagadnienie przedstawia problem rozwiązywania sudoku.

2.1 Struktury danych

W programie wykorzystano tablice do przechowywania wartości. W każdej komórce tabeli znajduje się wartość pola, informacja, czy została ona wpisana z pliku, oraz położenie pola w konsoli. W programie wykorzystano też listę do przechowywania wyników graczy. Takie struktury danych umożliwiają łatwy dostęp do zapisanych w nich elementów.

2.2 Algorytmy

Program rozwiązuje sudoku wstawiając kolejno cyfry od 1 do 9, w taki sposób, że żadna z cyfr nie powtarza się w wierszu, kolumnie i małym kwadracie. Najpierw sprawdza, czy dane pole jest puste, a następnie, czy wstawienie nowej cyfry nie będzie kolidowało z tymi, umieszczonymi w planszy wcześniej. Gdy wstawienie danej cyfry w taki sposób nie jest możliwe, program próbuje dopasować kolejną cyfrę. Jeśli cyfr nie da się ustawić w poprawny sposób program zwraca informacje, że sudoku nie da sie rozwiązać.[1].[2].

3 Specyfikacja zewnętrzna

Do uruchomienia programu nie jest wymagane podanie jakichkolwiek danych w linii poleceń. Program korzysta z plików wejściowych, które zawierają planszę (w tym też planszę z poprzedniej rozgrywki gracza). Może też istnieć plik, w którym zapisane są wyniki graczy. W przypadku, gdy pliki nie istnieją wyświetla się komunikat:

```
Blad otwarcia pliku.
```

Pliki są plikami tekstowymi.

4 Specyfikacja wewnętrzna

Program został zrealizowany zgodnie z paradygmatem strukturalnym. W programie rozdzielono interfejs (komunikację z użytkownikiem) od logiki aplikacji (rozwiazywanie sudoku).

4.1 Ogólna struktura programu

W funkcji głównej wywołana jest funkcja kolejny_stan . Funkcja ta obsługuje zmiany stanów. Gdy aktualny stan będzie STOP program kończy swoje działanie. Pierwszą wykonywaną funkcją jest do_menu, w której użytkownik wybiera kolejny stan programu. Nastepnie uruchamiana jest funkcja obsługująca wybrany przez użytkownika stan. Na przykład, gdy użytkownik wybierze opcję GRA uruchomiona zostaje funkcja do_gra, która, przez funkcje do_rozwiaz, tworzy planszę, umożliwia graczowi rozwiazanie, a nastepnie, dzięki funkcji do_sprawdz, umożliwia sprawdzenie poprawności rozwiązania. Gra kończy się, gdy sudoku jest poprawnie napisane i program przechodzi do stanu MENU.

4.2 Szczegółowy opis typów i funkcji

Szczegółowy opis typów i funkcji zawarty jest w załączniku.

5 Testowanie

Program został przetestowany na różnego rodzaju plikach. Pliki niepoprawne powodują zgłoszenie błędu. Plik zawierający planszę musi istnieć i być poprawny. W innym przypadku można korzystać tylko z funkcji, która tworzy nową planszę. Program został sprawdzony pod kątem wycieków pamięci.

6 Wnioski

Program Sudoku został stworzony wykorzystując programowanie obiektowe. Pozwoliło to na lepszą organizację w kodzie oraz ułatwiło dostęp do potrzebnych elementów. Początkowo sprawiało to trochę problemów, jednak z czasem udało się pokonać trudności. Wyzwaniem było także umieszczenie dynamicznych struktur w obiektach. Trzeba było pamiętać wtedy o prawidłowym zwalnianiu pamięci.

Literatura

[1] Adam Drozdek. C++. algorytmy i struktury danych. 2001.

[2] Jerzy Grębosz. *Symfonia C++*. Oficyna Kallimach, Kraków, 1995.

Dodatek
Szczegółowy opis typów i funkcji

Sudoku

Wygenerowano przez Doxygen 1.8.16

1 Indeks hierarchiczny 1

1.1 Hierarchia klas . . . 1

2 Indeks klas 3

2.1 Lista klas . . . 3

3 Dokumentacja klas 5

3.1 Dokumentacja klasy Gracz . . . 5

3.1.1 Opis szczegółowy . . . 5

3.1.2 Dokumentacja konstruktora i destruktora . . . 6

3.1.2.1 Gracz() . . . 6

3.1.2.2 ~Gracz() . . . 6

3.1.3 Dokumentacja funkcji składowych . . . 6

3.1.3.1 daj_czas() . . . 7

3.1.3.2 daj_nick() . . . 7

3.1.3.3 kolejny() . . . 7

3.1.3.4 ustaw() . . . 7

3.2 Dokumentacja klasy Plansza . . . 8

3.2.1 Opis szczegółowy . . . 8

3.2.2 Dokumentacja konstruktora i destruktora . . . 9

3.2.2.1 Plansza() [1/2] . . . 9

3.2.2.2 Plansza() [2/2] . . . 9

3.2.3 Dokumentacja funkcji składowych . . . 9

3.2.3.1 get_tab() . . . 10

3.2.3.2 operator=() . . . 11

3.2.3.3 set_tab() . . . 11

3.2.3.4 sprawdz() . . . 12

3.2.3.5 stworz_nowa() . . . 13

3.2.3.6 usun_losowe() . . . 13

3.2.3.7 wypisz_plansze() . . . 14

3.3 Dokumentacja klasy Pole . . . 15

3.3.1 Opis szczegółowy . . . 15

3.3.2 Dokumentacja konstruktora i destruktora . . . 16

3.3.2.1 Pole() [1/2] . . . 16

3.3.2.2 Pole() [2/2] . . . 16

3.3.3 Dokumentacja funkcji składowych . . . 16

3.3.3.1 get_wartosc() . . . 17

3.3.3.2 get_x() . . . 17

3.3.3.3 get_y() . . . 18

3.3.3.4 get_z_pliku() . . . 18

3.3.3.5 operator=() . . . 18

3.3.3.6 operator==() . . . 19

3.3.3.7 set_wartosc() . . . 19

3.3.3.8 set_z_pliku() . . . 20
3.3.3.9 wypisz() . . . 20
3.4 Dokumentacja klasy Rozwiaz1 . . . 20
3.4.1 Opis szczegółowy . . . 21
3.4.2 Dokumentacja funkcji składowych . . . 21
3.4.2.1 czy_moze() . . . 22
3.4.2.2 rozwiaz() . . . 22
3.4.2.3 uzyj_kolumna() . . . 23
3.4.2.4 uzyj_kwadrat() . . . 24
3.4.2.5 uzyj_wiersz() . . . 25
3.4.2.6 znajdz_puste() . . . 25
3.5 Dokumentacja klasy Rozwiaz2 . . . 26
3.5.1 Opis szczegółowy . . . 27
3.5.2 Dokumentacja funkcji składowych . . . 27
3.5.2.1 rozwiaz() . . . 27
3.6 Dokumentacja klasy Solver . . . 28
3.6.1 Opis szczegółowy . . . 29
3.6.2 Dokumentacja funkcji składowych . . . 29
3.6.2.1 rozwiaz_gracz() . . . 29
3.7 Dokumentacja klasy Stan . . . 30
3.7.1 Opis szczegółowy . . . 31
3.7.2 Dokumentacja konstruktora i destruktora . . . 31
3.7.2.1 Stan() . . . 32
3.7.3 Dokumentacja funkcji składowych . . . 32
3.7.3.1 do_gra() . . . 32
3.7.3.2 do_help() . . . 33
3.7.3.3 do_menu() . . . 33
3.7.3.4 do_rozwiaz() . . . 33
3.7.3.5 do_sprawdz() . . . 34
3.7.3.6 do_wygrana() . . . 34
3.7.3.7 do_wyjdz() . . . 34
3.7.3.8 do_wynik() . . . 35
3.7.3.9 do_zapisz() . . . 35
3.7.3.10 get_aktualny_stan() . . . 35
3.7.3.11 kolejny_stan() . . . 35
3.7.3.12 set_aktualny_stan() . . . 35
3.7.3.13 set_poprzedni_stan() . . . 36
3.8 Dokumentacja klasy Wygrana . . . 36
3.8.1 Opis szczegółowy . . . 37
3.8.2 Dokumentacja funkcji składowych . . . 37
3.8.2.1 kaczki() . . . 37
3.8.2.2 set_czas() . . . 38

3.8.2.3 wpisz_nick() . . . 38
3.8.2.4 zapisz() . . . 38
3.8.3 Dokumentacja przyjaciół i funkcji związanych . . . 39
3.8.3.1 operator<< . . . 39
3.9 Dokumentacja klasy Wynik . . . 40
3.9.1 Opis szczegółowy . . . 40
3.9.2 Dokumentacja funkcji składowych . . . 41
3.9.2.1 dodaj() . . . 41
3.9.2.2 operator+=() . . . 41
3.9.2.3 set_gracz() . . . 41
3.9.2.4 usun() . . . 42
3.9.2.5 wypisz_nick() . . . 42
3.9.2.6 wypisz_z_pliku() . . . 43
3.9.3 Dokumentacja przyjaciół i funkcji związanych . . . 44
3.9.3.1 operator>> . . . 44

Rozdział 1

Indeks hierarchiczny

1.1 Hierarchia klas

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

Gracz 5
Plansza 8
Pole 15
Solver 28
    Rozwiaz1 20
    Rozwiaz2 26
Stan 30
Wygrana 36
Wynik 40

Rozdział 2

Indeks klas

2.1 Lista klas

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

Gracz 5
Plansza 8
Pole 15
Rozwiaz1 20
Rozwiaz2 26
Solver 28
Stan 30
Wygrana 36
Wynik 40

Rozdział 3

Dokumentacja klas

3.1 Dokumentacja klasy Gracz

`#include <Wynik.h>`

Diagram współpracy dla Gracz:

Metody publiczne

- Gracz (Gracz ∗wsk, string nick_nowy, int czas_nowy)
- string daj_nick ()
- time_t daj_czas ()
- void ustaw (Gracz ∗z)
- Gracz ∗ kolejny ()
- ∼Gracz ()

Przyjaciele

- ostream & **operator**<< (ostream &out, Gracz ∗&)

3.1.1 Opis szczegółowy

Klasa Gracz - opisuje obiekt kazdego gracza

Parametry

pNext	wskaznik na kolejny element listy
nick	nick gracza
czas	czas gry gracza

Zobacz również

Gracz(Gracz∗ wsk, string nick_nowy, int czas_nowy)

string daj_nick()

time_t daj_czas()

void ustaw(Gracz∗ z)

Gracz∗ kolejny()

friend ostream& operator<<(ostream& out, Gracz∗&)

∼Gracz();

3.1.2 Dokumentacja konstruktora i destruktora

3.1.2.1 Gracz()

```
Gracz::Gracz (
            Gracz * wsk,
            string nick_nowy,
            int czas_nowy )  [inline]
```

Konstruktor obiektu Gracz

Parametry

wsk	wskaznik na kolejny element listy
nick_nowy	nick gracza
czas_nowy	czas gracza

3.1.2.2 ∼Gracz()

```
Gracz::~Gracz ( )
```

Destruktor obiektu Gracz

3.1.3 Dokumentacja funkcji składowych

3.1.3.1 daj_czas()

```
time_t Gracz::daj_czas ( )  [inline]
```

Funkcja wypisuje czas

Zwraca

Funkcja zwraca czas

3.1.3.2 daj_nick()

```
string Gracz::daj_nick ( )  [inline]
```

Funkcja wypisuje nick

Zwraca

Funkcja zwraca nick

3.1.3.3 kolejny()

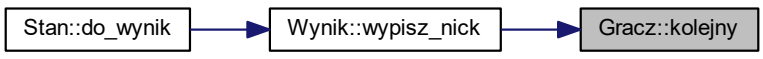
```
Gracz* Gracz::kolejny ( )  [inline]
```

Funkcja wypisuje wskaznik na kolejny element listy

Zwraca

Funkcja zwraca wskaznik na kolejny element listy

Oto graf wywoływań tej funkcji:



3.1.3.4 ustaw()

```
void Gracz::ustaw (
            Gracz * z )  [inline]
```

Funkcja wpisuje wskaznik na kolejny element listy

Parametry

z	wskaznik na kolejny element listy

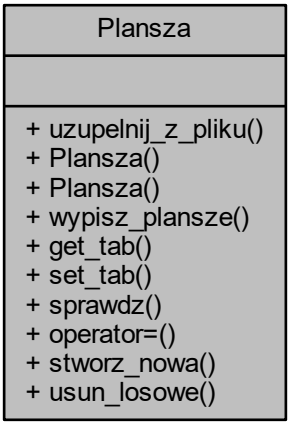
Dokumentacja dla tej klasy została wygenerowana z plików:

- Wynik.h
- Wyniki.cpp

3.2 Dokumentacja klasy Plansza

`#include <Plansza.h>`

Diagram współpracy dla Plansza:



Metody publiczne

- Pole **uzupelnij_z_pliku** (string)
- Plansza (Pole Tab)
- Plansza ()
- void wypisz_plansze ()
- Pole get_tab (int x, int y)
- Pole set_tab (int x, int y, Pole cos)
- bool sprawdz (Plansza)
- Plansza & operator= (Plansza &)
- Pole stworz_nowa (Plansza &)
- void usun_losowe (Plansza &)

3.2.1 Opis szczegółowy

Klasa Plansza opisuje plansze

Parametry

tab[9][9]	tablica obiektow Pole

Zobacz również

uzupelnij_z_pliku(string)

Plansza(Pole Tab)

Plansza()

wypisz_plansze()

Pole get_tab(int x, int y)

Pole set_tab(int x, int y, Pole cos)

bool sprawdz(Plansza)

Plansza& operator=(Plansza&)

Pole stworz_nowa(Plansza&)

void usun_losowe(Plansza&)

3.2.2 Dokumentacja konstruktora i destruktora

3.2.2.1 Plansza() [1/2]

```
Plansza::Plansza (
            Pole Tab ) [inline]
```

Konstruktor obiektu Pole

Parametry

Tab	tablica obiketow Pole

3.2.2.2 Plansza() [2/2]

```
Plansza::Plansza ( ) [inline]
```

Konstruktor obiektu Pole

3.2.3 Dokumentacja funkcji składowych

3.2.3.1 get_tab()

```
Pole Plansza::get_tab (
            int x,
            int y )
```

Funkcja wypisujaca element tablicy

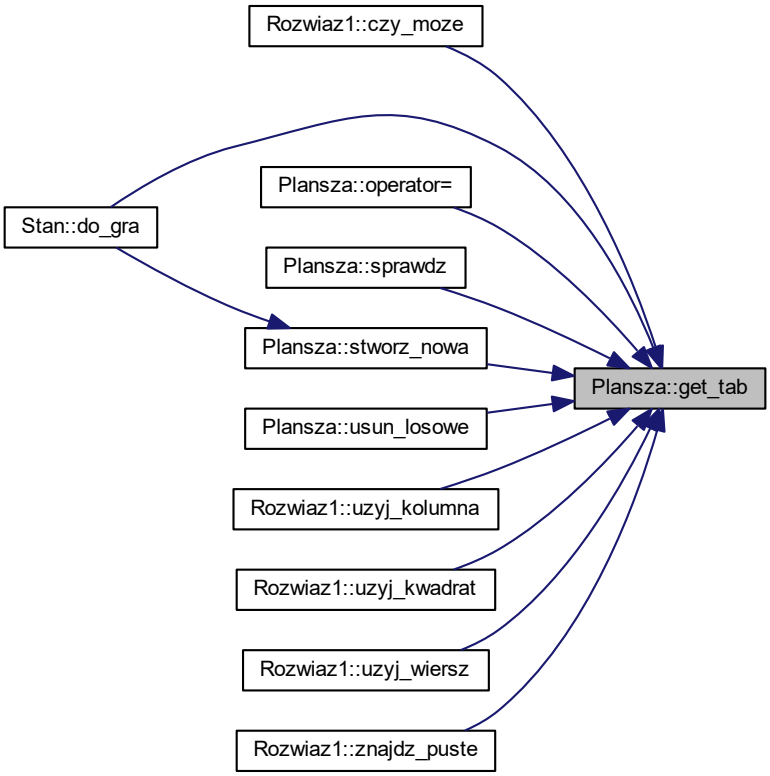
Parametry

x	wspolrzedne elementu tablicy
y	wspolrzedne elementu tablicy

Zwraca

Funkcja zwraca element Planszy

Oto graf wywoływań tej funkcji:



3.2.3.2 operator=()

```
Plansza & Plansza::operator= (
            Plansza & p )
```

Operator przypisania

Parametry

Plansza	nowa plansza

Zwraca

Operator zwraca plansze

Oto graf wywołań dla tej funkcji:

3.2.3.3 set_tab()

```
Pole Plansza::set_tab (
            int x,
            int y,
            Pole cos )
```

Funkcja wpisuje element tablicy

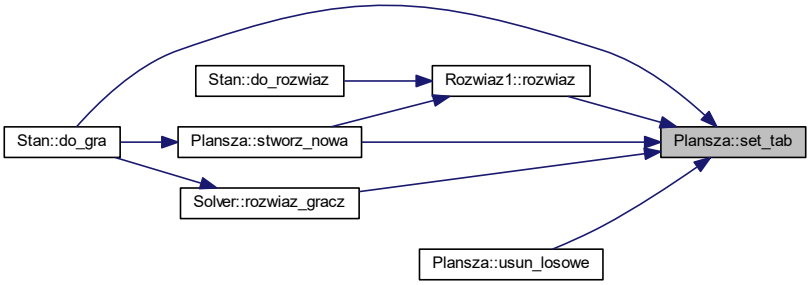
Parametry

x	wspolrzedne elementu tablicy
y	wspolrzedne elementu tablicy
cos	obiekt Pole do wpisania

Zwraca

Funkcja zwraca obiekt Pole

Oto graf wywoływań tej funkcji:



3.2.3.4 sprawdz()

```
bool Plansza::sprawdz (
            Plansza p )
```

Funkcja sprawdza poprawnosc wypelnienia planszy

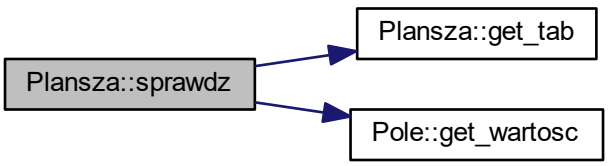
Parametry

Plansza	

Zwraca

Funkcja zwraca czy plansza jest poprawnie wypelniona

Oto graf wywołań dla tej funkcji:



3.2.3.5 stworz_nowa()

```
Pole Plansza::stworz_nowa (
            Plansza & plansza2 )
```

Funkcja tworzy nowa plansze

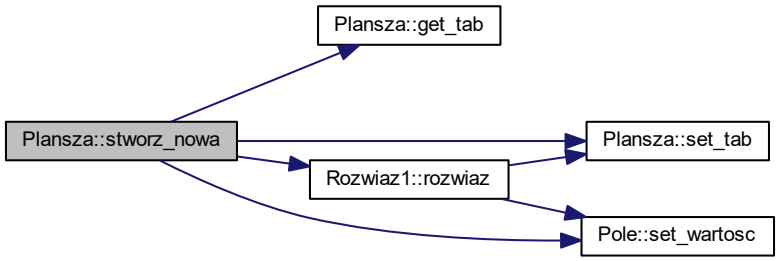
Parametry

Plansza	plansza do wypelnienia

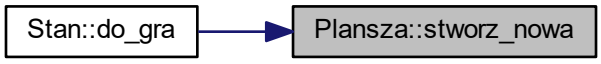
Zwraca

Funkcja zwraca poczatek nowej planszy

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



3.2.3.6 usun_losowe()

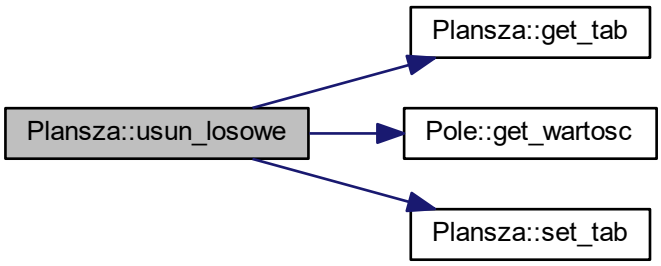
```
void Plansza::usun_losowe (
            Plansza & plansza2 )
```

Funkcja usuwa losowe wartosci z planszy

Parametry

Plansza	plansza z ktorej usuwamy wartosci

Oto graf wywołań dla tej funkcji:



3.2.3.7 wypisz_plansze()

```
void Plansza::wypisz_plansze ( )
```

Funkcja wypisujaca plansze Oto graf wywoływań tej funkcji:

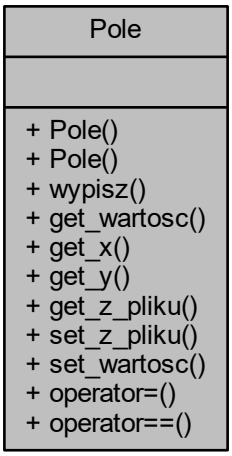
Dokumentacja dla tej klasy została wygenerowana z plików:

- Plansza.h
- Gra.cpp
- Menu.cpp
- Ogolne.cpp
- Sprawdz.cpp

3.3 Dokumentacja klasy Pole

`#include <Pole.h>`

Diagram współpracy dla Pole:



Metody publiczne

- Pole (int, bool, int, int)
- Pole ()
- void wypisz ()
- int get_wartosc ()
- int get_x ()
- int get_y ()
- bool get_z_pliku ()
- void set_z_pliku (bool nowa)
- void set_wartosc (int nowa)
- Pole & operator= (Pole &nowe)
- bool operator== (Pole)

3.3.1 Opis szczegółowy

Klasa Pole - pole planszy

Parametry

wartosc	wartosc pola
z_pliku	czy dana litera jest z pliku
x	polozenie pola w konsoli
y	polozenie pola w konsoli

Zobacz również

Pole(int, bool, int, int)

Pole()

void wypisz()

int get_wartosc()

int get_x()

int get_y()

bool get_z_pliku()

void set_z_pliku(bool nowa)

void set_wartosc(int nowa)

Pole& operator=(Pole& nowe)

bool operator==(Pole)

3.3.2 Dokumentacja konstruktora i destruktora

3.3.2.1 Pole() [1/2]

```
Pole::Pole (
            int w,
            bool z_p,
            int X,
            int Y )
```

Konstruktor obiektu Pole

Parametry

wartosc	wartosc pola
z_pliku	czy dana litera jest z pliku
x	polozenie pola w konsoli
y	polozenie pola w konsoli

3.3.2.2 Pole() [2/2]

```
Pole::Pole ( )  [inline]
```

Konstruktor obiektu Pole

3.3.3 Dokumentacja funkcji składowych

3.3.3.1 get_wartosc()

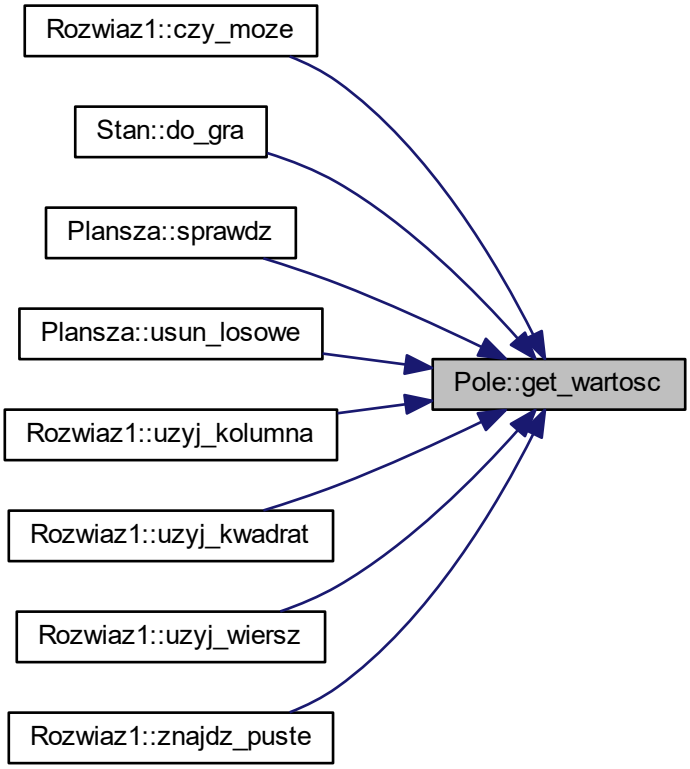
```
int Pole::get_wartosc ( )  [inline]
```

Funkcja wypisuje wartosc

Zwraca

Funkcja zwraca wartosc obiektu

Oto graf wywoływań tej funkcji:



3.3.3.2 get_x()

```
int Pole::get_x ( )  [inline]
```

Funkcja wypisujee polozenie pola

Zwraca

Funkcja zwraca polozenie x obiektu

3.3.3.3 get_y()

```
int Pole::get_y ( )  [inline]
```

Funkcja wypisuje polozenie pola

Zwraca

Funkcja zwraca polozenie y obiektu

3.3.3.4 get_z_pliku()

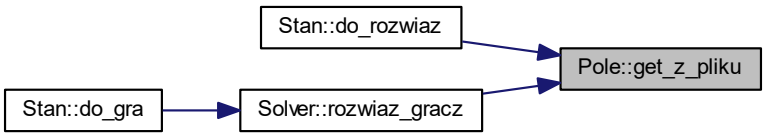
```
bool Pole::get_z_pliku ( )  [inline]
```

Funkcja wypisuje czy wartosc obiektu zostala wpisana z pliku

Zwraca

Funkcja zwraca czy wartosc obiektu zostala wpisana z pliku

Oto graf wywoływań tej funkcji:



3.3.3.5 operator=()

```
Pole & Pole::operator= (
            Pole & nowe )
```

Operator przypisania

Parametry

nowe	nowe pole

Zwraca

Operator zwraca pole

3.3.3.6 operator==()

```
bool Pole::operator== (
            Pole p )
```

Operator porownania

Parametry

Pole	pole do porownaina

Zwraca

Operator zwraca czy dane pola sa takie same

3.3.3.7 set_wartosc()

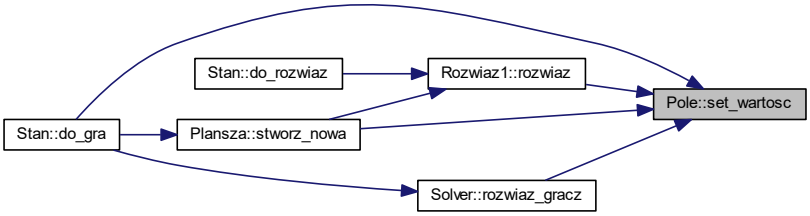
```
void Pole::set_wartosc (
            int nowa )  [inline]
```

Funkcja odczytuje wartosc pole

Parametry

nowa	nowa wartosc

Oto graf wywoływań tej funkcji:



3.3.3.8 set_z_pliku()

```
void Pole::set_z_pliku (
            bool nowa )  [inline]
```

Funkcja wpisuje czy wartosc pola pochodzi z pliku

Parametry

nowa	nowa wartosc

3.3.3.9 wypisz()

```
void Pole::wypisz ( )
```

Funkcja wypisuje pole

Dokumentacja dla tej klasy została wygenerowana z plików:

- Pole.h
- Menu.cpp
- Ogolne.cpp

3.4 Dokumentacja klasy Rozwiaz1

```
#include <Rozwiaz.h>
```

Diagram dziedziczenia dla Rozwiaz1

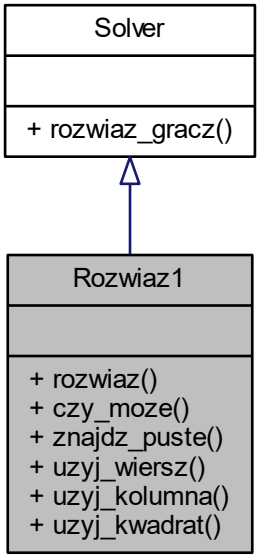
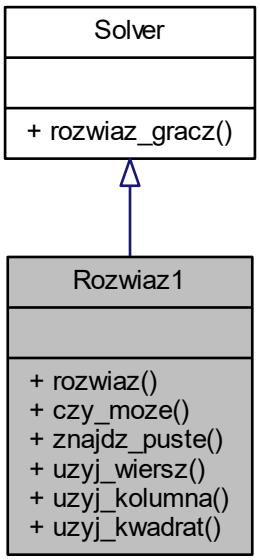


Diagram współpracy dla Rozwiaz1:



Metody publiczne

- bool rozwiaz (Plansza &)
- bool czy_moze (Plansza, int, int, int)
- bool znajdz_puste (Plansza, int &, int &)
- bool uzyj_wiersz (Plansza, int, int)
- bool uzyj_kolumna (Plansza, int, int)
- bool uzyj_kwadrat (Plansza, int, int, int)

3.4.1 Opis szczegółowy

Klasa Rozwiaz - klasa obslugujaca rozwiazywanie planszy

Zobacz również

bool rozwiaz(Plansza&)

bool czy_moze(Plansza, int, int, int)

bool znajdz_puste(Plansza, int&, int&)

bool uzyj_wiersz(Plansza, int, int)

bool uzyj_kolumna(Plansza, int, int)

bool uzyj_kwadrat(Plansza, int, int, int)

3.4.2 Dokumentacja funkcji składowych

3.4.2.1 czy_moze()

```
bool Rozwiaz1::czy_moze (
            Plansza plansza_stan,
            int wiersz,
            int kolumna,
            int i )
```

Funkcja sprawdza czy mozna umiescic cyfre

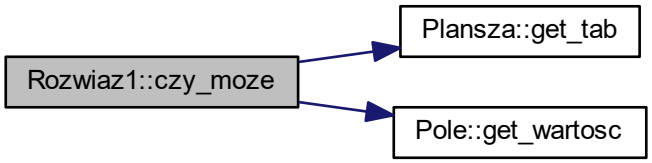
Parametry

Plansza	plansza
int	wiersz
int	kolumna
int	cyfra

Zwraca

Funkcja zwraca czy mozna umiescic cyfre

Oto graf wywołań dla tej funkcji:



3.4.2.2 rozwiaz()

```
bool Rozwiaz1::rozwiaz (
            Plansza & plansza_stan )
```

Funkcja umozliwiajaca automatyczne rozwiazywanie sudoku

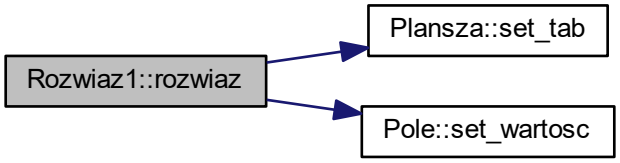
Parametry

Plansza	plansza

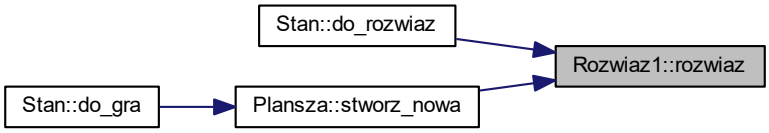
Zwraca

Funkcja zwraca czy plansza zostala rozwiazana

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



3.4.2.3 uzyj_kolumna()

```
bool Rozwiaz1::uzyj_kolumna (
            Plansza plansza_stan,
            int kolumna,
            int nr )
```

Funkcja sprawdza czy mozna umiescic cyfre w kolumnie

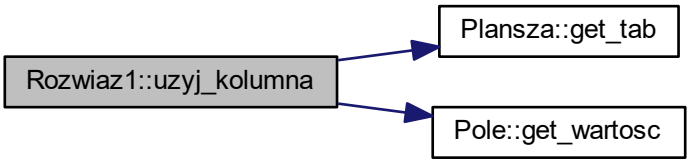
Parametry

Plansza	plansza
int	kolumna
int	cyfra

Zwraca

Funkcja zwraca czy mozna umiescic cyfre w kolumnie

Oto graf wywołań dla tej funkcji:



3.4.2.4 uzyj_kwadrat()

```
bool Rozwiaz1::uzyj_kwadrat (
            Plansza plansza_stan,
            int wiersz,
            int kolumna,
            int nr )
```

Funkcja sprawdza czy mozna umiescic cyfre w kwadracie

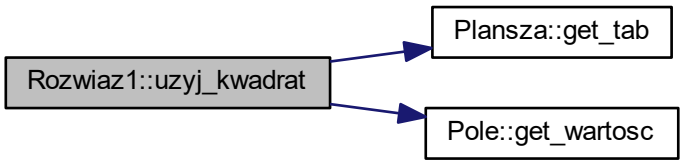
Parametry

Plansza	plansza
int	wiersz
int	kolumna
int	cyfra

Zwraca

Funkcja zwraca czy mozna umiescic cyfre w kwadracie

Oto graf wywołań dla tej funkcji:



3.4.2.5 uzyj_wiersz()

```
bool Rozwiaz1::uzyj_wiersz (
            Plansza plansza_stan,
            int wiersz,
            int nr )
```

Funkcja sprawdza czy mozna umiescic cyfre w wierszu

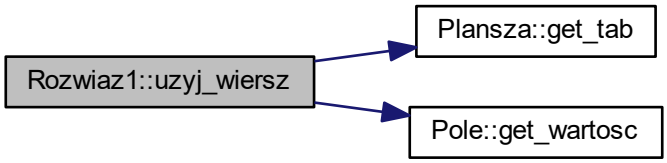
Parametry

Plansza	plansza
int	wiersz
int	cyfra

Zwraca

Funkcja zwraca czy mozna umiescic cyfre w wierszu

Oto graf wywołań dla tej funkcji:



3.4.2.6 znajdz_puste()

```
bool Rozwiaz1::znajdz_puste (
            Plansza plansza_stan,
            int & wiersz,
            int & kolumna )
```

Funkcja znajduje puste pole

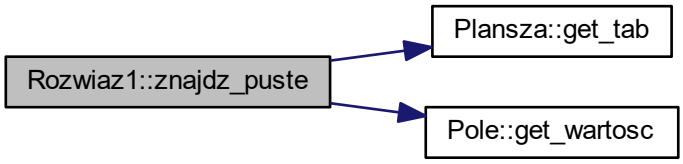
Parametry

Plansza	plansza
int	wiersz
int	kolumna

Zwraca

Funkcja zwraca czy pole jest puste

Oto graf wywołań dla tej funkcji:



Dokumentacja dla tej klasy została wygenerowana z plików:

- Rozwiaz.h
- Rozwiaz.cpp

3.5 Dokumentacja klasy Rozwiaz2

`#include <Rozwiaz.h>`

Diagram dziedziczenia dla Rozwiaz2

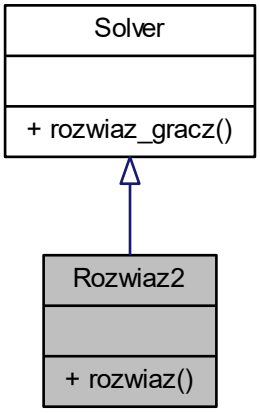
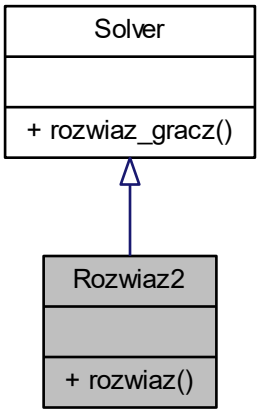


Diagram współpracy dla Rozwiaz2:



Metody publiczne

- bool rozwiaz (Plansza &)

3.5.1 Opis szczegółowy

Klasa Rozwiaz2 - klasa obslugujaca inna metode rozwiazywania sudoku

Zobacz również

bool rozwiaz(Plansza&)

3.5.2 Dokumentacja funkcji składowych

3.5.2.1 rozwiaz()

```
bool Rozwiaz2::rozwiaz (
            Plansza &  )  [inline]
```

Funkcja umozliwiajaca automatyczne rozwiazywanie sudoku, ktora kiedys moze zostanie dodana

Parametry

Plansza	plansza

Zwraca

Funkcja zwraca czy plansza zostala rozwiazana

Dokumentacja dla tej klasy została wygenerowana z pliku:

- Rozwiaz.h

3.6 Dokumentacja klasy Solver

`#include <Rozwiaz.h>`

Diagram dziedziczenia dla Solver

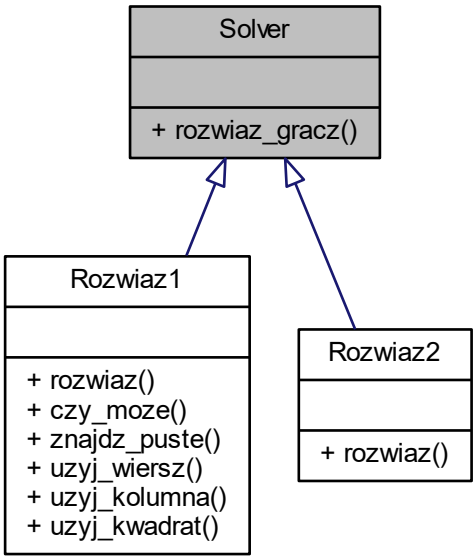
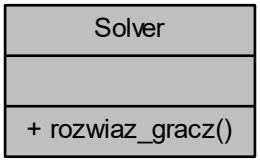


Diagram współpracy dla Solver:



Metody publiczne

- state rozwiaz_gracz (Plansza &)

3.6.1 Opis szczegółowy

Klasa Solver - klasa bazowa

Zobacz również

state rozwiaz_gracz(Plansza&)

3.6.2 Dokumentacja funkcji składowych

3.6.2.1 rozwiaz_gracz()

```
state Solver::rozwiaz_gracz (
            Plansza & p )
```

Funkcja umozliwiajaca rozwiazanie sudoku przez gracza

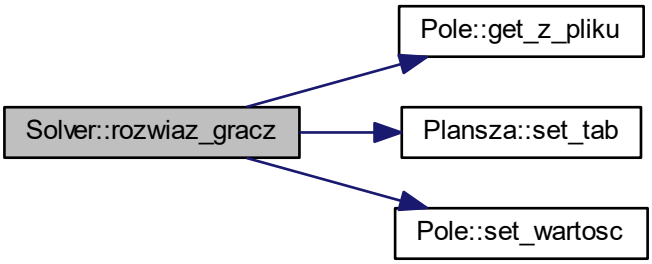
Parametry

Plansza	plansza

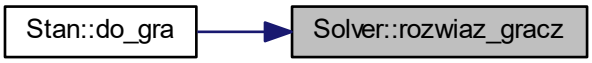
Zwraca

Funkcja zwraca kolejny stan

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



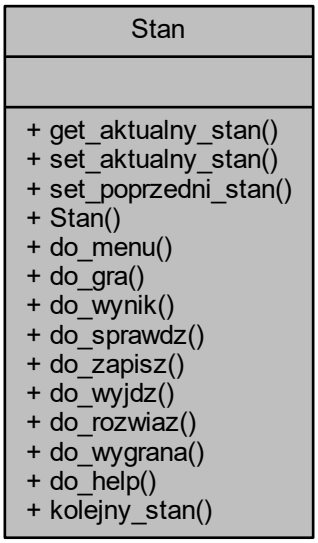
Dokumentacja dla tej klasy została wygenerowana z plików:

- Rozwiaz.h
- Rozwiaz.cpp

3.7 Dokumentacja klasy Stan

`#include <Stan.h>`

Diagram współpracy dla Stan:



Metody publiczne

- state get_aktualny_stan ()
- void set_aktualny_stan (state stan)
- void set_poprzedni_stan (state stan)
- Stan (state stan)
- state do_menu (void)
- state do_gra ()
- state do_wynik ()
- state do_sprawdz ()
- state do_zapisz ()
- state do_wyjdz ()
- state do_rozwiaz ()
- state do_wygrana ()
- state do_help ()
- void kolejny_stan ()

3.7.1 Opis szczegółowy

Klasa Stan - obsluguje zmiany stanow

Parametry

aktualny_stan	aktualny stan
plansza_stan	plansza
poprzedni_stan	poprzedni stan
czas	czas rozgrywki
tab[NUMBER]	tablica wskaznikow na funkcje obslugujace stany

Zobacz również

state get_aktualny_stan()

void set_aktualny_stan(state stan)

void set_poprzedni_stan(state stan)

Stan(state stan)

state do_menu(void)

state do_gra()

state do_wynik()

state do_sprawdz()

state do_zapisz()

state do_wyjdz()

state do_rozwiaz()

state do_wygrana()

state do_help()

void kolejny_stan()

3.7.2 Dokumentacja konstruktora i destruktora

3.7.2.1 Stan()

```
Stan::Stan (
            state stan )  [inline]
```

Konstruktor obiektu Stan

Parametry

stan	aktualny stan

3.7.3 Dokumentacja funkcji składowych

3.7.3.1 do_gra()

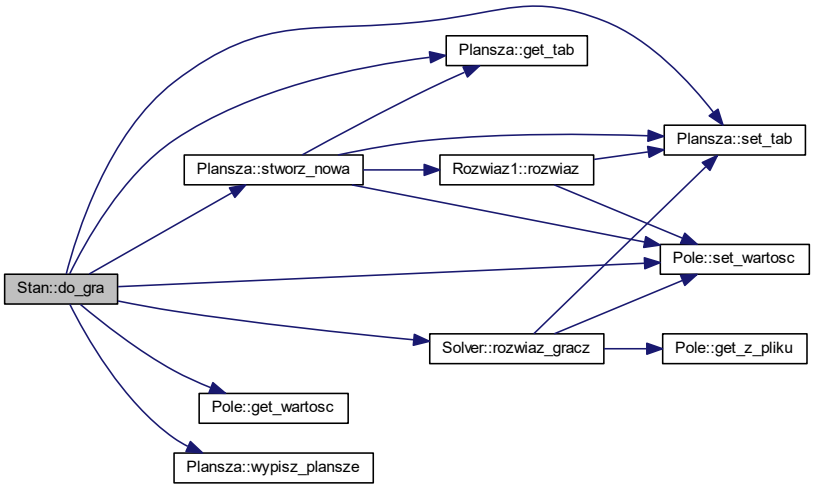
```
state Stan::do_gra ( )
```

Funkcja obslugujaca stan GRA

Zwraca

Funkcja zwraca kolejny stan

Oto graf wywołań dla tej funkcji:



3.7.3.2 do_help()

```
state Stan::do_help ( )
```

Funkcja obslugujaca stan HELP

Zwraca

Funkcja zwraca kolejny stan

3.7.3.3 do_menu()

```
state Stan::do_menu (
            void  )
```

Funkcja obslugujaca stan MENU

Zwraca

Funkcja zwraca kolejny stan

3.7.3.4 do_rozwiaz()

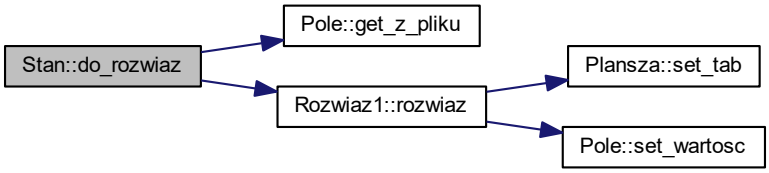
```
state Stan::do_rozwiaz ( )
```

Funkcja obslugujaca stan ROZWIAZ

Zwraca

Funkcja zwraca kolejny stan

Oto graf wywołań dla tej funkcji:



3.7.3.5 do_sprawdz()

```
state Stan::do_sprawdz ( )
```

Funkcja obslugujaca stan SPRAWDZ

Zwraca

Funkcja zwraca kolejny stan

3.7.3.6 do_wygrana()

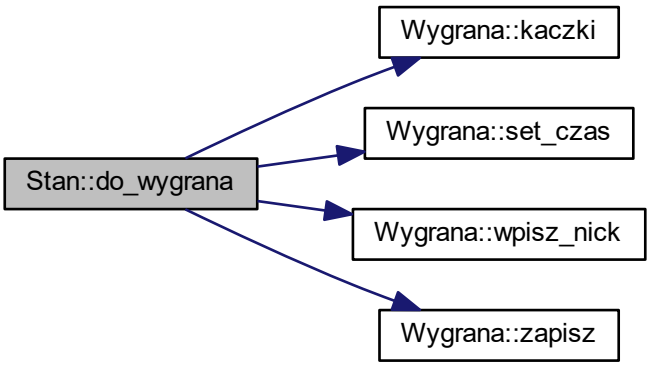
```
state Stan::do_wygrana ( )
```

Funkcja obslugujaca stan WYGRANA

Zwraca

Funkcja zwraca kolejny stan

Oto graf wywołań dla tej funkcji:



3.7.3.7 do_wyjdz()

```
state Stan::do_wyjdz ( )
```

Funkcja obslugujaca stan WYJDZ

Zwraca

Funkcja zwraca kolejny stan

3.7.3.8 do_wynik()

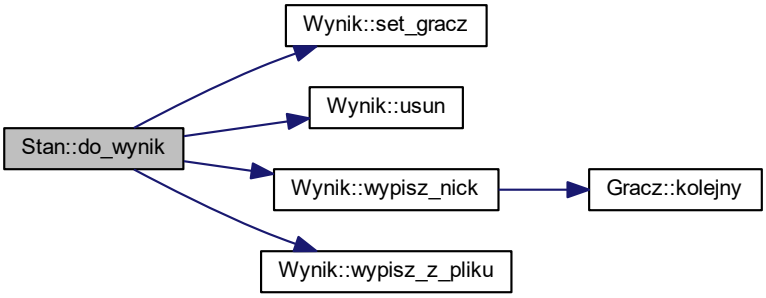
```
state Stan::do_wynik ( )
```

Funkcja obslugujaca stan WYNIK

Zwraca

Funkcja zwraca kolejny stan

Oto graf wywołań dla tej funkcji:



3.7.3.9 do_zapisz()

```
state Stan::do_zapisz ( )
```

Funkcja obslugujaca stan ZAPISZ

Zwraca

Funkcja zwraca kolejny stan

3.7.3.10 get_aktualny_stan()

```
state Stan::get_aktualny_stan ( )
```

Funkcja wypisujaca aktualny stan @retrun Funkcja zwraca aktualny stan

3.7.3.11 kolejny_stan()

```
void Stan::kolejny_stan ( )
```

Funkcja ustawia kolejny stan

3.7.3.12 set_aktualny_stan()

```
void Stan::set_aktualny_stan (
            state stan )
```

Funkcja wpisujaca aktualny stan

Parametry

stan	aktualny stan

3.7.3.13 set_poprzedni_stan()

```
void Stan::set_poprzedni_stan (
            state stan )  [inline]
```

Funkcja wpisujaca poprzedni stan

Parametry

stan	poprzedni stan

Dokumentacja dla tej klasy została wygenerowana z plików:

- Stan.h
- Gra.cpp
- Menu.cpp
- Ogolne.cpp
- Rozwiaz.cpp
- Sprawdz.cpp
- Sudoku.cpp
- Wygrana.cpp
- Wyniki.cpp
- Zapisz.cpp

3.8 Dokumentacja klasy Wygrana

```
#include <Wygrana.h>
```

Diagram współpracy dla Wygrana:

Metody publiczne

- void kaczki ()
- void set_czas (time_t nowy)
- bool zapisz ()
- void wpisz_nick ()

Przyjaciele

- ostream & operator<< (ostream &out, Wygrana &w)

3.8.1 Opis szczegółowy

Klasa Wygrana - klasa obslugujaca stan wygrana

Parametry

nick	nick gracza

Zobacz również

void kaczki()

void set_czas(time_t nowy)

bool zapisz(time_t)

void wpisz_nick()

friend ostream& operator<<(ostream& out, Wygrana & w)

3.8.2 Dokumentacja funkcji składowych

3.8.2.1 kaczki()

```
void Wygrana::kaczki ( )
```

Funkcja wypisujaca kaczki Oto graf wywoływań tej funkcji:

3.8.2.2 set_czas()

```
void Wygrana::set_czas (
            time_t nowy )  [inline]
```

Funkcja wpisujaca czas gry

Parametry

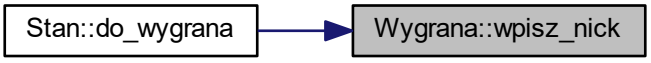
nowy	nowy czas

Oto graf wywoływań tej funkcji:

3.8.2.3 wpisz_nick()

```
void Wygrana::wpisz_nick ( )
```

Funkcja wpisujaca nick Oto graf wywoływań tej funkcji:



3.8.2.4 zapisz()

```
bool Wygrana::zapisz ( )
```

Funkcja zapisujaca wynik do pliku

Parametry

time_t	czas rozgrywki

Zwraca

Funkcja zwraca czy dane zostaly prawidlowo zapisane

Oto graf wywoływań tej funkcji:

3.8.3 Dokumentacja przyjaciół i funkcji związanych

3.8.3.1 operator<<

```
ostream& operator<< (
            ostream & out,
            Wygrana & w )  [friend]
```

Operator strumieniowy

Parametry

out	wyjscie
w	obiekt klasy Wygrana

Zwraca

Funkcja zwraca wyjscie

Dokumentacja dla tej klasy została wygenerowana z plików:

- Wygrana.h
- Wygrana.cpp

3.9 Dokumentacja klasy Wynik

`#include <Wynik.h>`

Diagram współpracy dla Wynik:

Metody publiczne

- void set_gracz (Gracz ∗nowy)
- void dodaj (Gracz ∗&pHead, string nick_nowy, int czas_nowy)
- void usun ()
- void wypisz_nick ()
- bool wypisz_z_pliku ()
- Gracz ∗ operator+= (Gracz ∗)

Przyjaciele

- istream & operator>> (istream &input, Wynik &w)

3.9.1 Opis szczegółowy

Klasa Wynik - poczatek listy wynikow

Parametry

pHead	poczatek listy wynikow

Zobacz również

void set_gracz(Gracz∗ nowy)

void dodaj(Gracz∗& pHead, string nick_nowy, int czas_nowy)

void usun(Gracz∗& pHead)

void wypisz_nick(Gracz* pHead)

bool wypisz_z_pliku(Gracz*&)

3.9.2 Dokumentacja funkcji składowych

3.9.2.1 dodaj()

```
void Wynik::dodaj (
            Gracz *& pHead,
            string nick_nowy,
            int czas_nowy )
```

Funkcja dodaje nowy element do listy

Parametry

pHead	wskaznik na poczatek listy
nick_nowy	nick gracza
czas_nowy	czas gracza

3.9.2.2 operator+=()

```
Gracz * Wynik::operator+= (
            Gracz * x )
```

Operator +=

Parametry

Gracz	wskaznik na obiekt Gracz

Zwraca

Operator zwraca wskaznik na Gracz

3.9.2.3 set_gracz()

```
void Wynik::set_gracz (
            Gracz * nowy )  [inline]
```

Funkcja wpisuje nowy poczatek listy graczy

Parametry

nowy	nowy wskaznik

Oto graf wywoływań tej funkcji:

3.9.2.4 usun()

```
void Wynik::usun ( )
```

Funkcja usuwa liste wynikow

Parametry

pHead	wskaznik na poczatek listy

Oto graf wywoływań tej funkcji:

3.9.2.5 wypisz_nick()

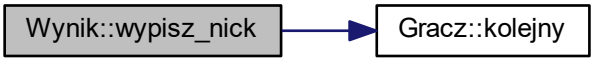
```
void Wynik::wypisz_nick ( )
```

Funkcja wypisuje liste wynikow

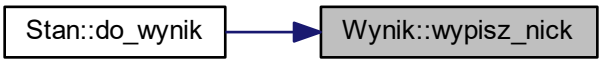
Parametry

pHead	wskaznik na poczatek listy

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



3.9.2.6 wypisz_z_pliku()

```
bool Wynik::wypisz_z_pliku ( )
```

Funkcja wypisuje dane z pliku

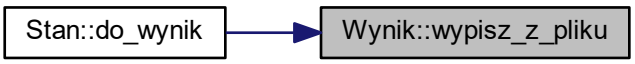
Parametry

wskaznik	na poczatek listy

Zwraca

Funkcja zwraca czy dane zostaly prawidlowo zapisane

Oto graf wywoływań tej funkcji:



3.9.3 Dokumentacja przyjaciół i funkcji związanych

3.9.3.1 operator>>

```
istream& operator>> (
            istream & input,
            Wynik & w )  [friend]
```

Operator strumieniowy

Parametry

input	wejscie
w	obiekt klasy Wygrana

Zwraca

Funkcja zwraca wejscie

Dokumentacja dla tej klasy została wygenerowana z plików:

- Wynik.h
- Wyniki.cpp