Politechnika Śląska
Wydział Informatyki, Elektroniki i Informatyki

Programowanie Komputerów 2

Sudoku

autor	Sylwia Molitor
prowadzący	mgr inż. Wojciech Dudzik
rok akademicki	2018/2019
kierunek	informatyka
rodzaj studiów	SSI
semestr	2
termin laboratorium	wtorek, 12:00 – 13:30
sekcja	72
termin oddania sprawozdania	2019-06-24
ścieżka dostępu do pliku na dysku sieciowym	github

1 Treść zadania

Gra Sudoku jest rozgrywana na planszy 9x9. Zadaniem użytkownika jest wypełnienie planszy tak, aby cyfry od 1 do 9 nie powtarzały się w mniejszych kwadratach, poziomo i pionowo. Zadaniem komputera może być sprawdzenie poprawności wypełnienia planszy przez użytkownika jak i samodzielne wypełnienie. Program wczytuje z pliku tekstowego planszę i może zapisywać lub odczytywać stan gry. Użytkownik podaje współrzędne miejsca, w które chce wpisać liczbę. Nie można nadpisywać wartości, występują podpowiedzi, poziomy trudności, generator plansz i w przypadku przerwania program zapisuje się. Do wyboru są trzy początkowe plansze - użytkownik może tworzyć nowe.

2 Algorytmy, struktury danych, ograniczenia specyfikacji

W programie wykorzystano strukturę złożoną z dwóch tablic, jednej przechowującej dane liczbowe, a drugiej informację o możliwości nadpisania danego miejsca. Jest to struktura dynamiczna.

Algorytmy wykorzystane w programie dotyczą: wypełniania sudoku, sprawdzania poprawności wypełnienia, generacji nowej planszy poprzez losowanie pierwszego elementu i wypełnianie pozostałej części tablicy, po czym usunięciu liczby elementów ze względu na poziom trudności.

Ograniczenia specyfikacji dotyczą głównie tego, że użytkownik nie widzi, które cyfry może nadpisać. Dopiero po próbie nadpisu wyświetla się stosowny komunikat o braku możliwości dodania lub cyfra zostaje dodana.

2.1 Analiza problemu, podstawy teoretyczne

Sudoku to gra polegająca na wypełnianiu tablicy o rozmiarach 9x9 cyframi od 1 do 9. Cyfry nie mogą się powtarzać w kolumnach, wierszach, ani w małuch kwadratach 3x3.

3 Specyfikacja zewnętrzna

3.1 Obsługa programu

Program jest uruchamiany z linii poleceń. Należy przekazać do programu nazwę programu oraz nazwę pliku wejściowego. Przykład:

```
projekt plikTekstowy1.txt
```

Pliki są plikami tekstowymi, ale mogą mieć dowolne rozszerzenie (pliki wejściowe muszą mieć podczas uruchamiania wpisane rozszerzenie .txt). Jeśli plik źródłowy jest pusty lub nie istnieje to przy pierwszym uruchomieniu programu należy wygenerować planszę. Podanie nieprawidłowej nazwy pliku powoduje, że program tworzy nowy plik bez wygenerowanej planszy - użytkownik musi sam ją wygenerować. Po uruchomieniu programu wykorzystuje się skróty klawiszowe, które są opisane w menu programu: a, aby wczytać z pliku ostatnią grę, g, aby wygenerować grę, r, tryb rozwiązywania , s, aby sprawdzić, h, aby uzyskać 1 podpowiedź, c , aby komputer rozwiązał, z, aby zakończyć.

W trybie rozwiązywania wprowadza się współrzędne punktu talbicy, w które chce się dodać cyfrę, a następnie jej wartość. W trybie generacji gry, wprowadza się cyfrę, która przemnożona przez 10 określi, ile cyfr zostanie usuniętych z planszy.

3.2 Format danych wejściowych

W pliku wejściowym musi znajdować się tablica sudoku (9 linii cyfr lub spacji po 9 znaków plus znaki końca linii), oddzielonych znakiem nowej linii od tablicy określającej, czy można nadpisywać cyfry (9 linii 0 lub 1 po 10 znaków oddzielonych znakami nowej linii). Jeśli plik ma niewłaściwe dane, należy wygenerować nową tablicę sudoku, aby program spełniał swoją funkcję.

3.3 Komunikaty

Podczas rozgrywki wywoływane są różnego rodzaju komunikaty. Mówią one o prawidłowym, bądź złym wypełnienia planszy, braku miejsca do dopisywania znaków, braku możliwości nadpisania wartości, a także precyzują co użytkownik ma zrobić w danym momencie.

4 Specyfikacja wewnętrzna - zmienne i funkcje

Zmienne i funkcje są szczegółowo opisane na końcu, w pliku wygenerowanym przez Doxygen. W funkcji głównej rezerwowana jest pamięć i wywołana jest funkcja

```
contactWithUser().
```

W niej przebiega komunikacja użytkownika z program (na początku zostają wczytane dane z pliku). Jej składowymi są główne funkcje: generujące -

`generateBoard()`,

wypełniające -

`writeInTheBoard()`,

rozwiązujące -

`solve_it()`

i sprawdzające planszę -

`checking()`.

Po wyjściu z tej funkcji, zostają wywołane funkcje czyszczenia -

`cleaning()`

i zapisywania do pliku -

`saving()`,

po czym zostaje zwolniona pamięć. Najczęściej używane zmienne w programie to

`pointer`

i

`pointerBO`.

Wskazują one odpowiednio na tablicę cyfr i tablicę określającą, czy można nadpisywać cyfry.

5 Testowanie

Program działa dla prawidłowo wprowadzonych danych. W przeciwnym razie należy wygenerować tablicę sudoku w programie. Do projektu zostały załączone 3 przykładowe plansze o zróżnicowanym poziomie trudności.

6 Wnioski

Praca nad programem była bardzo czasochłonna. Dużo czasu zajęło opracowanie w czytelny sposób menu porozumiewania się z użytkownikiem.

Dodatek
Szczegółowy opis typów i funkcji

My Project

Generated by Doxygen 1.8.15

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

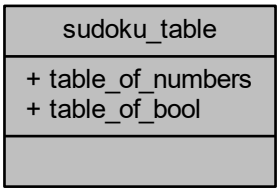
Chapter 3

Data Structure Documentation

3.1 sudoku_table Struct Reference

`#include <struct.h>`

Collaboration diagram for sudoku_table:



Data Fields

- char table_of_numbers [90]

  tablica liczb w sudoku
- bool table_of_bool [90]

  tablica bool (ktora zawiera informacje, czy mozna nadpisac cyfre)

3.1.1 Detailed Description

Tablica sudoku

3.1.2 Field Documentation

3.1.2.1 table_of_bool

```
bool table_of_bool[90]
```

tablica bool (ktora zawiera informacje, czy mozna nadpisac cyfre)

3.1.2.2 table_of_numbers

```
char table_of_numbers[90]
```

tablica liczb w sudoku

The documentation for this struct was generated from the following file:

- struct.h

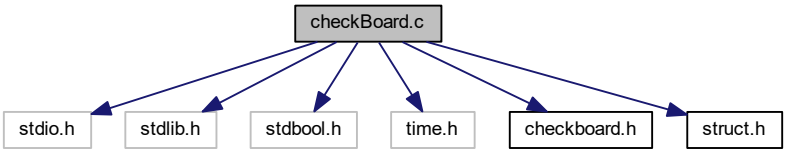
Chapter 4

File Documentation

4.1 checkBoard.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <time.h>
#include "checkboard.h"
#include "struct.h"
```

Include dependency graph for checkBoard.c:



Functions

- bool checking (char *pointer)
- bool checkingColumns (char *pointer)
- bool checkingRows (char *pointer)
- bool checkingSquares (char *pointer)

4.1.1 Function Documentation

4.1.1.1 checking()

```
bool checking (
            char * pointer )
```

Funkcja sprawdza, czy tablica sudoku jest dobrze wypelniona

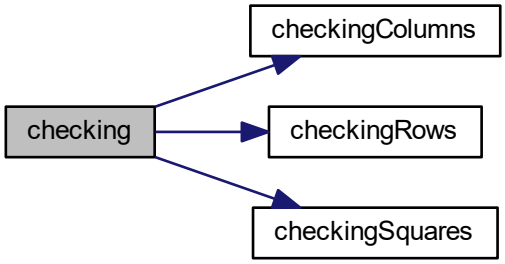
Parameters

pointer	wskaznik na tablice sudoku

Returns

true, gdy prawidlowo wypelnione
false, gdy zle wypelnione

Here is the call graph for this function:



4.1.1.2 checkingColumns()

```
bool checkingColumns (
            char * pointer )
```

Funkcja sprawdza, czy kolumny sudoku sa dobrze wypelnione

Parameters

pointer	wskaznik na tablice sudoku

Returns

true, gdy prawidlowo wypelnione
false, gdy zle wypelnione

4.1.1.3 checkingRows()

```
bool checkingRows (
            char * pointer )
```

Funkcja sprawdza, czy wiersze sudoku sa dobrze wypelnione

Parameters

pointer	wskaznik na tablice sudoku

Returns

true, gdy prawidlowo wypelnione
false, gdy zle wypelnione

4.1.1.4 checkingSquares()

```
bool checkingSquares (
            char * pointer )
```

Funkcja sprawdza, czy kwadraty w sudoku sa dobrze wypelnione

Parameters

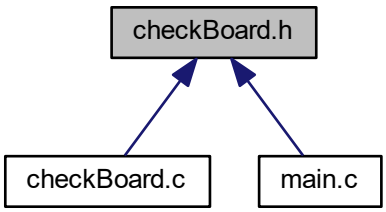
pointer	wskaznik na tablice sudoku

Returns

true, gdy prawidlowo wypelnione
false, gdy zle wypelnione

4.2 checkBoard.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define _CHECKBOARD_H

Functions

- bool checking (char ∗pointer)
- bool checkingColumns (char ∗pointer)
- bool checkingRows (char ∗pointer)
- bool checkingSquares (char ∗pointer)

4.2.1 Macro Definition Documentation

4.2.1.1 _CHECKBOARD_H

```
#define _CHECKBOARD_H
```

4.2.2 Function Documentation

4.2.2.1 checking()

```
bool checking (
            char * pointer )
```

Funkcja sprawdza, czy tablica sudoku jest dobrze wypelniona

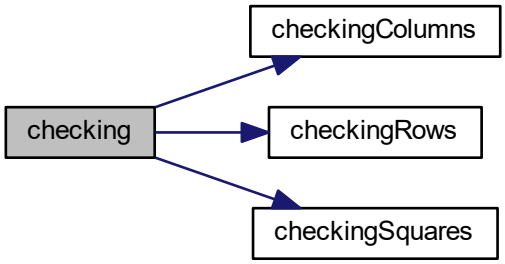
Parameters

pointer	wskaznik na tablice sudoku

Returns

true, gdy prawidlowo wypelnione
false, gdy zle wypelnione

Here is the call graph for this function:



4.2.2.2 checkingColumns()

```
bool checkingColumns (
            char * pointer )
```

Funkcja sprawdza, czy kolumny sudoku sa dobrze wypelnione

Parameters

pointer	wskaznik na tablice sudoku

Returns

true, gdy prawidlowo wypelnione
false, gdy zle wypelnione

4.2.2.3 checkingRows()

```
bool checkingRows (
            char * pointer )
```

Funkcja sprawdza, czy wiersze sudoku sa dobrze wypelnione

Parameters

pointer	wskaznik na tablice sudoku

Returns

true, gdy prawidlowo wypelnione
false, gdy zle wypelnione

4.2.2.4 checkingSquares()

```
bool checkingSquares (
            char * pointer )
```

Funkcja sprawdza, czy kwadraty w sudoku sa dobrze wypelnione

Parameters

pointer	wskaznik na tablice sudoku

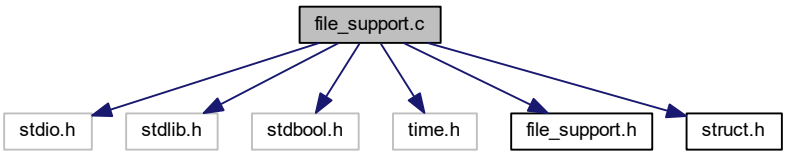
Returns

true, gdy prawidlowo wypelnione
false, gdy zle wypelnione

4.3 file_support.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <time.h>
#include "file_support.h"
#include "struct.h"
```

Include dependency graph for file_support.c:



Functions

- void reading (char *pointer, bool *pointerBO, char *number_of_file)
- void saving (char *pointer, bool *pointerBO, char *number_of_file)
- void cleaning (char *number_of_file)

4.3.1 Function Documentation

4.3.1.1 cleaning()

```
void cleaning (
            char * number_of_file )
```

Funkcja czysci plik do zapisu

Parameters

number_of_file	nazwa pliku

4.3.1.2 reading()

```
void reading (
            char * pointer,
            bool * pointerBO,
            char * number_of_file )
```

Funkcja odczytuje z pliku przykladowe sudoku

Parameters

pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number_of_file	nazwa pliku

4.3.1.3 saving()

```
void saving (
            char * pointer,
            bool * pointerBO,
            char * number_of_file )
```

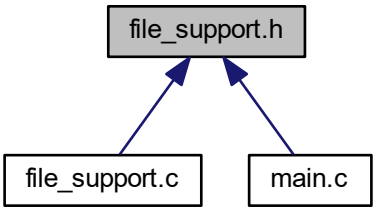
Funkcja zapisuje do pliku przykladowe sudoku

Parameters

pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number_of_file	nazwa pliku

4.4 file_support.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define _FILE_SUPPORT_H

Functions

- void reading (char ∗pointer, bool ∗pointerBO, char ∗number_of_file)
- void saving (char ∗pointer, bool ∗pointerBO, char ∗number_of_file)
- void cleaning (char ∗number_of_file)

4.4.1 Macro Definition Documentation

4.4.1.1 _FILE_SUPPORT_H

```
#define _FILE_SUPPORT_H
```

4.4.2 Function Documentation

4.4.2.1 cleaning()

```
void cleaning (
            char * number_of_file )
```

Funkcja czysci plik do zapisu

Parameters

number_of_file	nazwa pliku

4.4.2.2 reading()

```
void reading (
             char * pointer,
             bool * pointerBO,
             char * number_of_file )
```

Funkcja odczytuje z pliku przykladowe sudoku

Parameters

pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number_of_file	nazwa pliku

4.4.2.3 saving()

```
void saving (
             char * pointer,
             bool * pointerBO,
             char * number_of_file )
```

Funkcja zapisuje do pliku przykladowe sudoku

Parameters

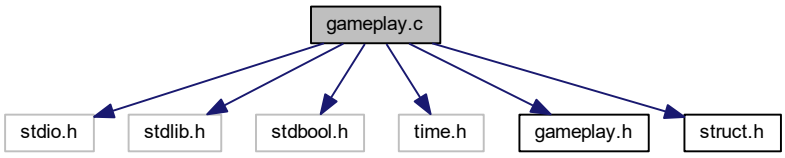
pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number_of_file	nazwa pliku

4.5 gameplay.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <time.h>
#include "gameplay.h"
```

```
#include "struct.h"
```

Include dependency graph for gameplay.c:



Functions

- void deleting (char ∗pointer, bool ∗pointerBO, char number)
- void write_out (char ∗pointer)
- void writeInTheBoard (char ∗pointer, int vertical, int horizontal, char number, bool ∗pointerBO)
- void generateBoard (char ∗pointer, bool ∗pointerBO, char number)
- bool check_empty_spaces (char ∗pointer)
- void contactWithUser (char ∗pointer, bool ∗pointerBO, char ∗number_of_file)

4.5.1 Function Documentation

4.5.1.1 check_empty_spaces()

```
bool check_empty_spaces (
            char * pointer )
```

Funkcja sprawdza, czy sa jakies puste miejsca w sudoku

Parameters

pointer	wskaznik na tablice sudoku

Returns

true gdy istnieja wolne miejsca
false gdy nie istnieja wolne miejsca

4.5.1.2 contactWithUser()

```
void contactWithUser (
            char * pointer,
```

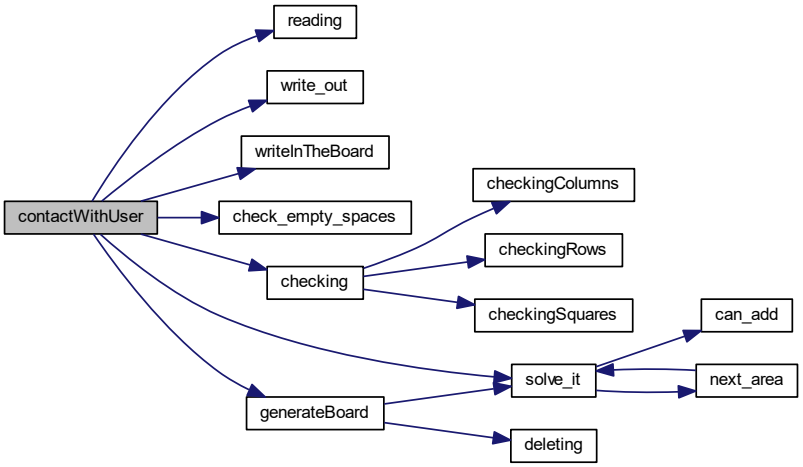
```
            bool * pointerBO,
            char * number_of_file )
```

Funkcja umozliwia kontakt z uzytkownikiem

Parameters

pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number_of_file	nazwa pliku

Here is the call graph for this function:



4.5.1.3 deleting()

```
void deleting (
            char * pointer,
            bool * pointerBO,
            char number )
```

Funkcja usuwa 10 razy wiecej cyfr od podanej od uzytkownika liczby z wygenerowanej tablicy sudoku

Parameters

pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number	liczba podana przez uzytkownika

4.5.1.4 generateBoard()

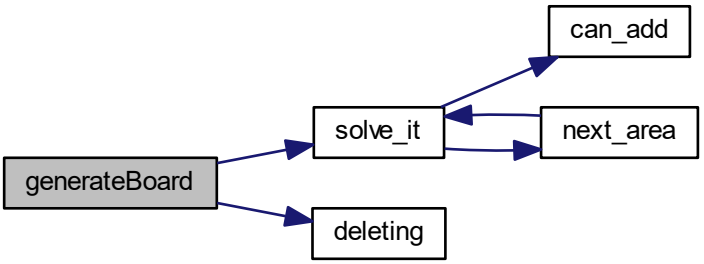
```
void generateBoard (
            char * pointer,
            bool * pointerBO,
            char number )
```

Funkcja generuje przykladowe sudoku o poziomie trudnosci podanym przez uzytkownika

Parameters

pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number	liczba podana przez uzytkownika

Here is the call graph for this function:



4.5.1.5 write_out()

```
void write_out (
            char * pointer )
```

Funkcja wypisuje na ekranie konsoli sudoku

Parameters

pointer	wskaznik na tablice sudoku

4.5.1.6 writeInTheBoard()

```
void writeInTheBoard (
```

```
char * pointer,
int vertical,
int horizontal,
char number,
bool * pointerBO )
```

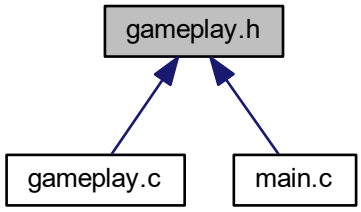
Funkcja wpisuje podana przez uzytkownika cyfre do danego miejsca

Parameters

pointer	wskaznik na tablice sudoku
vertical	pionowa wspolrzedna miejsca wpisania cyfry
horizontal	pozioma wspolrzedna miejsca wpisania cyfry
number	liczba podana przez uzytkownika
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)

4.6 gameplay.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define _GAMEPLAY_H

Functions

- void deleting (char ∗pointer, bool ∗pointerBO, char number)
- void write_out (char ∗pointer)
- void writeInTheBoard (char ∗pointer, int vertical, int horizontal, char number, bool ∗pointerBO)
- void generateBoard (char ∗pointer, bool ∗pointerBO, char number)
- bool check_empty_spaces (char ∗pointer)
- void contactWithUser (char ∗pointer, bool ∗pointerBO, char ∗number_of_file)

4.6.1 Macro Definition Documentation

4.6.1.1 _GAMEPLAY_H

```
#define _GAMEPLAY_H
```

4.6.2 Function Documentation

4.6.2.1 check_empty_spaces()

```
bool check_empty_spaces (
            char * pointer )
```

Funkcja sprawdza, czy sa jakies puste miejsca w sudoku

Parameters

pointer	wskaznik na tablice sudoku

Returns

true gdy istnieja wolne miejsca
false gdy nie istnieja wolne miejsca

4.6.2.2 contactWithUser()

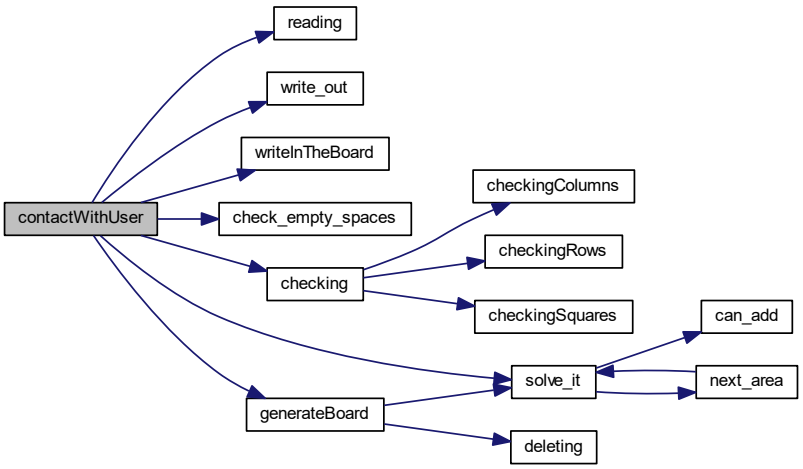
```
void contactWithUser (
            char * pointer,
            bool * pointerBO,
            char * number_of_file )
```

Funkcja umozliwia kontakt z uzytkownikiem

Parameters

pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number_of_file	nazwa pliku

Here is the call graph for this function:



4.6.2.3 deleting()

```
void deleting (
            char * pointer,
            bool * pointerBO,
            char number )
```

Funkcja usuwa 10 razy wiecej cyfr od podanej od uzytkownika liczby z wygenerowanej tablicy sudoku

Parameters

pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number	liczba podana przez uzytkownika

4.6.2.4 generateBoard()

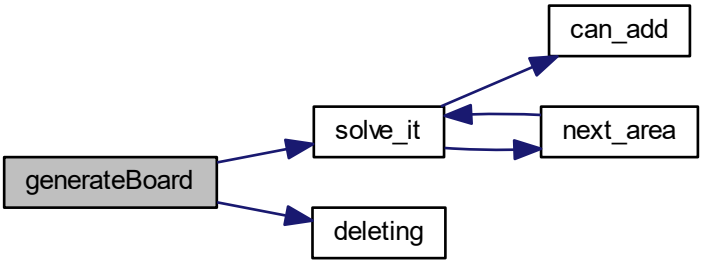
```
void generateBoard (
            char * pointer,
            bool * pointerBO,
            char number )
```

Funkcja generuje przykladowe sudoku o poziomie trudnosci podanym przez uzytkownika

Parameters

pointer	wskaznik na tablice sudoku
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)
number	liczba podana przez uzytkownika

Here is the call graph for this function:



4.6.2.5 write_out()

```
void write_out (
            char * pointer )
```

Funkcja wypisuje na ekranie konsoli sudoku

Parameters

pointer	wskaznik na tablice sudoku

4.6.2.6 writeInTheBoard()

```
void writeInTheBoard (
            char * pointer,
            int vertical,
            int horizontal,
            char number,
            bool * pointerBO )
```

Funkcja wpisuje podana przez uzytkownika cyfre do danego miejsca

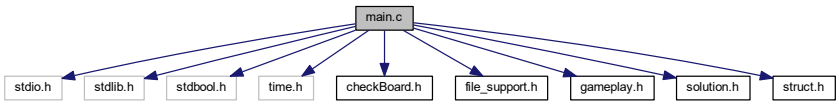
Parameters

pointer	wskaznik na tablice sudoku
vertical	pionowa wspolrzedna miejsca wpisania cyfry
horizontal	pozioma wspolrzedna miejsca wpisania cyfry
number	liczba podana przez uzytkownika
pointerBO	wskaznik na tablice bool(ktora zawiera informacje czy mozna nadpisac cyfre)

4.7 main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <time.h>
#include "checkBoard.h"
#include "file_support.h"
#include "gameplay.h"
#include "solution.h"
#include "struct.h"
```

Include dependency graph for main.c:



Functions

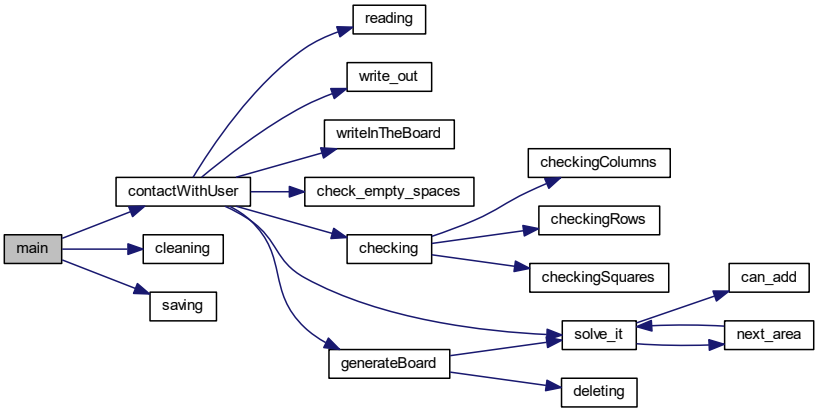
- int main (int argc, char ∗∗argv)

4.7.1 Function Documentation

4.7.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

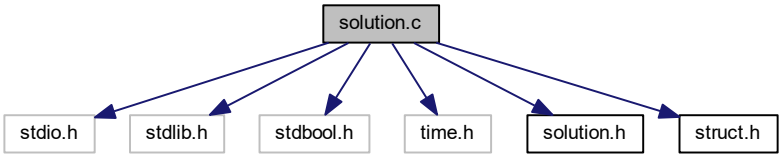
Here is the call graph for this function:



4.8 solution.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <time.h>
#include "solution.h"
#include "struct.h"
```

Include dependency graph for solution.c:



Functions

- bool can_add (int x, int y, int value, char ∗pointer, char ∗helping_pointer)
- bool next_area (int x, int y, char ∗pointer, char ∗helping_pointer)
- bool solve_it (int x, int y, char ∗pointer, char ∗helping_pointer)

4.8.1 Function Documentation

4.8.1.1 can_add()

```
bool can_add (
             int x,
             int y,
             int value,
             char * pointer,
             char * helping_pointer )
```

Funkcja sprawdza, czy mozna dodac dana cyfre w dane miejsce

Parameters

x	wspolrzedna x miejsca dodania
y	wspolrzedna y miejsca dodania
value	cyfra do dodania
pointer	wskaznik na tablice sudoku
helping_pointer	wskaznik na pomocnicza tablice

Returns

true jesli mozna dodac
false jesli nie mozna dodac

4.8.1.2 next_area()

```
bool next_area (
             int x,
             int y,
             char * pointer,
             char * helping_pointer )
```

Funkcja umozliwia przejscie w nastepne miejsce tablicy sudoku

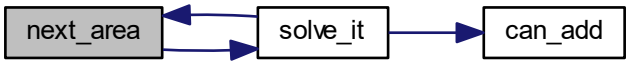
Parameters

x	wspolrzedna x miejsca dodania
y	wspolrzedna y miejsca dodania
pointer	wskaznik na tablice sudoku
helping_pointer	wskaznik na pomocnicza tablice

Returns

true jesli koniec tablicy

Here is the call graph for this function:



4.8.1.3 solve_it()

```
bool solve_it (
            int x,
            int y,
            char * pointer,
            char * helping_pointer )
```

Funkcja wypelnia sudoku

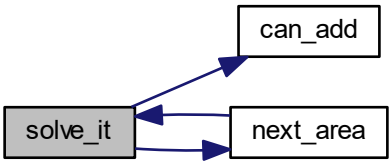
Parameters

x	wspolrzedna x miejsca dodania
y	wspolrzedna y miejsca dodania
pointer	wskaznik na tablice sudoku
helping_pointer	wskaznik na pomocnicza tablice

Returns

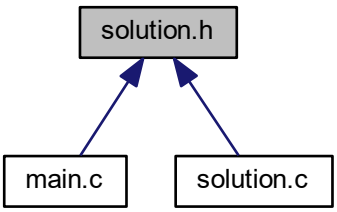
true jesli dodano element
false jesli nie dodano elementu

Here is the call graph for this function:



4.9 solution.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define _SOLUTION_H

Functions

- bool can_add (int x, int y, int value, char ∗pointer, char ∗helping_pointer)
- bool next_area (int x, int y, char ∗pointer, char ∗helping_pointer)
- bool solve_it (int x, int y, char ∗pointer, char ∗helping_pointer)

4.9.1 Macro Definition Documentation

4.9.1.1 _SOLUTION_H

```
#define _SOLUTION_H
```

4.9.2 Function Documentation

4.9.2.1 can_add()

```
bool can_add (
            int x,
            int y,
            int value,
            char * pointer,
            char * helping_pointer )
```

Funkcja sprawdza, czy mozna dodac dana cyfre w dane miejsce

Parameters

x	wspolrzedna x miejsca dodania
y	wspolrzedna y miejsca dodania
value	cyfra do dodania
pointer	wskaznik na tablice sudoku
helping_pointer	wskaznik na pomocnicza tablice

Returns

true jesli mozna dodac
false jesli nie mozna dodac

4.9.2.2 next_area()

```
bool next_area (
            int x,
            int y,
            char * pointer,
            char * helping_pointer )
```

Funkcja umozliwia przejscie w nastepne miejsce tablicy sudoku

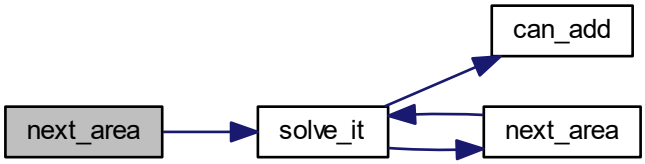
Parameters

x	wspolrzedna x miejsca dodania
y	wspolrzedna y miejsca dodania
pointer	wskaznik na tablice sudoku
helping_pointer	wskaznik na pomocnicza tablice

Returns

true jesli koniec tablicy

Here is the call graph for this function:



4.9.2.3 solve_it()

```
bool solve_it (
            int x,
            int y,
            char * pointer,
            char * helping_pointer )
```

Funkcja wypelnia sudoku

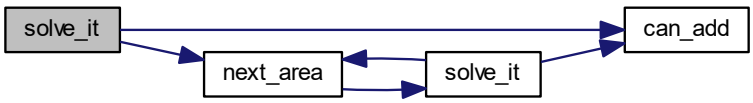
Parameters

x	wspolrzedna x miejsca dodania
y	wspolrzedna y miejsca dodania
pointer	wskaznik na tablice sudoku
helping_pointer	wskaznik na pomocnicza tablice

Returns

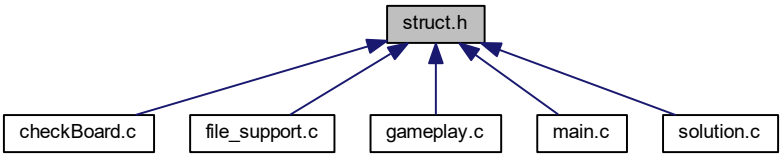
true jesli dodano element
false jesli nie dodano elementu

Here is the call graph for this function:



4.10 struct.h File Reference

This graph shows which files directly or indirectly include this file:



Data Structures

- struct sudoku_table

Macros

- #define _STRUCT_H

4.10.1 Macro Definition Documentation

4.10.1.1 _STRUCT_H

```
#define _STRUCT_H
```

Index