

My Project

Wygenerowano przez Doxygen 1.8.15

1 Indeks struktur danych	1
1.1 Struktury danych	1
2 Indeks plików	3
2.1 Lista plików	3
3 Dokumentacja struktur danych	5
3.1 Dokumentacja struktury Ekwipunekstr	5
3.2 Dokumentacja struktury kordy	5
3.3 Dokumentacja struktury pole	6
3.4 Dokumentacja struktury tabwyn	6
3.5 Dokumentacja struktury wrog	6
4 Dokumentacja plików	7
4.1 Dokumentacja pliku edytormap.h	7
4.1.1 Dokumentacja funkcji	7
4.1.1.1 edycja()	7
4.1.1.2 edytormap()	8
4.1.1.3 nowamapa()	8
4.1.1.4 przemieszczenie()	8
4.1.1.5 staramapa()	8
4.1.1.6 wktorastrone()	8
4.1.1.7 wpisz()	9
4.2 Dokumentacja pliku gra.h	9
4.2.1 Dokumentacja funkcji	9
4.2.1.1 dodlista()	9
4.2.1.2 gra()	10
4.2.1.3 klawiatura()	10
4.2.1.4 otwdrzwi()	10
4.2.1.5 poruszanie()	12
4.2.1.6 ruch()	12
4.2.1.7 znajdzewkw()	13
4.2.1.8 znaku()	13
4.3 Dokumentacja pliku Header.h	14
4.3.1 Dokumentacja funkcji	15
4.3.1.1 czyszczenie()	15
4.3.1.2 czyszczeniebufora()	15
4.3.1.3 dowolnyprzycisk()	15
4.3.1.4 gotoxy()	15
4.3.1.5 podmien()	15
4.4 Dokumentacja pliku obsługapluku.h	16
4.4.1 Dokumentacja funkcji	16
4.4.1.1 czytaniemapy()	16

4.4.1.2 zapiszmape()	17
4.4.1.3 zczytaj()	17
4.4.1.4 zczytajtablice()	18
4.5 Dokumentacja pliku przeciwnicy.h	18
4.5.1 Dokumentacja funkcji	18
4.5.1.1 dodprzec()	19
4.5.1.2 gdzieiscprze()	20
4.5.1.3 poruszprzeciwnikami()	20
4.5.1.4 potworprzed()	21
4.5.1.5 ruchnormalny()	21
4.5.1.6 ruchnormalnyzw()	22
4.5.1.7 ruchodwrotny()	22
4.5.1.8 ruchodwrotnyzw()	22
4.5.1.9 usunprzeciwi()	23
4.5.1.10 wyszczscruch()	23
4.5.1.11 zmianapozycjiprze()	24
4.5.1.12 znajdzprzeciwnika()	24
4.6 Dokumentacja pliku tablicawynikow.h	24
4.6.1 Dokumentacja funkcji	25
4.6.1.1 dodajwynik()	25
4.6.1.2 tablicawynikow()	25
4.6.1.3 wypiszdotablicy()	25
4.6.1.4 zapisztablice()	26
4.7 Dokumentacja pliku usuwanie.h	26
4.7.1 Dokumentacja funkcji	26
4.7.1.1 usuncalamape()	26
4.7.1.2 usunekwipunek()	27
4.7.1.3 usunliste()	27
4.7.1.4 usunmape()	27
4.7.1.5 usunnazwe()	28
4.7.1.6 usuntablice()	28
4.8 Dokumentacja pliku wypisywanie.h	28
4.8.1 Dokumentacja funkcji	28
4.8.1.1 wypiszekw()	28
4.8.1.2 wypiszmapa()	29
4.8.1.3 wypiszmapiekw()	29
4.8.1.4 wypiszpomchodzenie()	29
4.8.1.5 wypiszpomoc()	30
Indeks	31

Rozdział 1

Indeks struktur danych

1.1 Struktury danych

Tutaj znajdują się struktury danych wraz z ich krótkimi opisami:

Ekwipunekstr	5
kordy	5
pole	6
tabwyn	6
wrog	6

Rozdział 2

Indeks plików

2.1 Lista plików

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

edytormap.h	7
gra.h	9
Header.h	14
obsługapliku.h	16
przeciwnicy.h	18
tablicawynikow.h	24
usuwanie.h	26
wypisywanie.h	28

Rozdział 3

Dokumentacja struktur danych

3.1 Dokumentacja struktury Ekwipunekstr

Pola danych

- struct [Ekwipunekstr](#) * [pNext](#)
wskaznik na kolejny element listy
- unsigned char [przedmiot](#)
rodzaj przedmiotu

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [Header.h](#)

3.2 Dokumentacja struktury kordy

Pola danych

- int [cordx](#)
pozycja w pionie
- int [cordy](#)
pozycja w poziomie

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [Header.h](#)

3.3 Dokumentacja struktury pole

Pola danych

- unsigned char [obraz](#)
jaki obrazek ma sie wyswietlac na tym polu
- int [wejście](#)
czy gracz lub potwor moze wejsc na to pole

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [Header.h](#)

3.4 Dokumentacja struktury tabwyt

Pola danych

- struct [tabwyt](#) * [pNext](#)
wskaznik na kolejny element tablicy
- char [nazwa](#) [30]
dlugosc nazwy gracza
- int [wynik](#)
ilosc punktow, jaka zdobyl gracz

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [Header.h](#)

3.5 Dokumentacja struktury wrog

Pola danych

- [kordy pozycja](#)
pozycja wroga na mapie
- int [kierunek](#)
kierunek, w ktorym sie porusza
- unsigned char [naczymstoi](#)
co znajduje sie na polu na ktorym stoi
- struct [wrog](#) * [pNext](#)
wskaznik na kolejnego przeciwnika w liscie
- bool [ruch](#)
czy przeciwnik sie ruszyl w tym przebiegu petli

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [Header.h](#)

Rozdział 4

Dokumentacja plików

4.1 Dokumentacja pliku edytormap.h

```
#include "Header.h"
```

Funkcje

- void `edytormap` ()
- void `nowamapa` ()
- void `staramapa` ()
- void `wpisz` (int znak, `kordy` pozycja, `wskpole` mapa)
- int `wktoastrone` ()
- void `przemieszczenie` (`kordy` *pozycja, int wiersze, int kolumny)
- void `edycja` (`wskpole` mapa, int wiersze, int kolumny, `wrog` **przeciwnicy, `kordy` **postac)

4.1.1 Dokumentacja funkcji

4.1.1.1 edycja()

```
void edycja (
    wskpole mapa,
    int wiersze,
    int kolumny,
    wrog ** przeciwnicy,
    kordy ** postac )
```

Funkcja pozwala edytować mapę oraz ją zapisać

Parametry

<i>mapa</i>	wskaznik na mapę
<i>wiersze</i>	ilość wierszy mapy
<i>kolumny</i>	ilość kolumn mapy
<i>przeciwnicy</i>	wskaznik na listę przeciwników
<i>pozycja</i>	postacie na mapie

4.1.1.2 edytormap()

```
void edytormap ( )
```

Funkcja obsługuje uruchomienie edytora map

4.1.1.3 nowamapa()

```
void nowamapa ( )
```

Funkcja tworzy nowa mape

4.1.1.4 przemieszczenie()

```
void przemieszczenie (
    kordy * pozycja,
    int wiersze,
    int kolumny )
```

Funkcja obsługuje poruszenie się po mapie podczas edycji

Parametry

<i>pozycja</i>	aktualna pozycja znacznika
<i>wiersze</i>	ilość wierszy mapy
<i>kolumny</i>	ilość kolumn mapy

4.1.1.5 staramapa()

```
void staramapa ( )
```

Funkcja wczytuje istniejącą mapę

4.1.1.6 wktorastrone()

```
int wktorastrone ( )
```

Funkcja czytuje od gracza, w którą stronę ma poruszać się przeciwnik

Zwraca

kierunek poruszania się

4.1.1.7 wpisz()

```
void wpisz (
    int znak,
    kordy pozycja,
    wskpole mapa )
```

Funkcja zapisuje znak do tablicy

Parametry

<i>znak</i>	jak iznak ma byc wpisany
<i>pozycja</i>	w jakim miejscu ma sie znajdowac ten znak
<i>mapa</i>	wskaznik na mape

4.2 Dokumentacja pliku gra.h

```
#include "Header.h"
```

Funkcje

- void **gra** (*tabwyn* **pHead, char *nazwa)
- bool **klawiatura** (*kordy* *ja, *pole* ***tablica, *Ekwipunekstr* **pGlowa, int wiersze, int kolumny, *wrog* **przeciw)
- bool **poruszanie** (*kordy* *ja, *kordy* *kolja, *pole* ***tablica, *Ekwipunekstr* **pGlowa, int wiersze, int kolumny, unsigned char *tymczasowy, bool *koniec)
- void **dodlista** (*Ekwipunekstr* **pGlowa, unsigned char *znak)
- void **znaklu** (int wysokosc, *Ekwipunekstr* **pGlowa, unsigned char *znak)
- *Ekwipunekstr* * **znajdzewkw** (*Ekwipunekstr* *pGlowa, unsigned char znak)
- bool **otwdrzwi** (int wysokosc, *Ekwipunekstr* **pGlowa, unsigned char *znak)
- bool **ruch** (*kordy* *ja, *kordy* *kolja)

4.2.1 Dokumentacja funkcji

4.2.1.1 dodlista()

```
void dodlista (
    Ekwipunekstr ** pGlowa,
    unsigned char * znak )
```

Funkcja dodaje nowy klucz do listy ekwipunku

Parametry

<i>pGlowa</i>	wskaznik na liste kluczy ktore posiada gracz
<i>znak</i>	jaki klucz jest wstawiany

4.2.1.2 gra()

```
void gra (
    tabwyn ** pHead,
    char * nazwa )
```

Funkcja obsługuje uruchomienie gry oraz jej zakończenie

Parametry

<i>pHead</i>	wskaznik na tablice wynikow
<i>nazwa</i>	imie uzytkownika

4.2.1.3 klawiatura()

```
bool klawiatura (
    kordy * ja,
    pole *** tablica,
    Ekwipunekstr ** pGlowa,
    int wiersze,
    int kolumny,
    wrog ** przeciw )
```

Funkcja czysci konsrole

Parametry

<i>ja</i>	wskaznik na pozycje gracza
<i>tablica</i>	wskaznik na tablice zawierajaca mape
<i>pGlowa</i>	wskaznik na tablice wynikow
<i>wiersze</i>	ilosc wierszy mapy
<i>kolumny</i>	ilosc kolumn mapy
<i>przeciw</i>	wspaznik na liste przeciwnikow

Zwraca

czy gracz wygral (true - wygral, false - przegral)

4.2.1.4 otwdrzwi()

```
bool otwdrzwi (
    int wysokosc,
```

```
Ekwipunekstr ** pGlowa,  
unsigned char * znak )
```

Funkcja usuwa klucz z ekwipunku i otwiera drzwi

Parametry

<i>wysokosc</i>	ilosc wierszy mapy
<i>pGlowa</i>	wskaznik na liste kluczy ktore posiada gracz
<i>jake</i>	drzwi gracz probuje otworzyc

Zwraca

czy udalo sie otworzyc drzwi

4.2.1.5 poruszanie()

```
bool poruszanie (
    kordy * ja,
    kordy * kolja,
    pole *** tablica,
    Ekwipunekstr ** pGlowa,
    int wiersze,
    int kolumny,
    unsigned char * tymczasowy,
    bool * koniec )
```

Funkcja obsluguje poruszenie gracza

Parametry

<i>ja</i>	wskaznik na pozycje gracza
<i>kolja</i>	wskaznik na nastepna pozycje gracza
<i>tablica</i>	wskaznik na tablice zawierajaca mape
<i>pGlowa</i>	wskaznik na liste kluczy ktore posiada gracz
<i>wiersze</i>	ilosc wierszy mapy
<i>kolumny</i>	ilosc kolumn mapy
<i>tymczasowy</i>	wskaznik na znak jaki miało pole przed wejściem gracza
<i>koniec</i>	czy gracz doszedł do wyjścia z mapy

Zwraca

wartość sygnału wyjściowego

4.2.1.6 ruch()

```
bool ruch (
    kordy * ja,
    kordy * kolja )
```

Funkcja czytuje wciśnięty klawisz oraz przekazuje kolejną pozycję gracza

Parametry

<i>ja</i>	wskaznik na pozycje gracza
<i>kolja</i>	wskaznik na nastepna pozycje gracza

Zwraca

czy gracz chce wyjsc z rozgrywki

4.2.1.7 `znajdzwekw()`

```
Ekwipunekstr* znajdzwekw (
    Ekwipunekstr * pGlowa,
    unsigned char znak )
```

Funkcja znajduje klucz w ekwipunku

Parametry

<i>pGlowa</i>	wskaznik na liste kluczy ktore posiada gracz
<i>znak</i>	jaki klucz ma byc zaleziony w ekwipunku

Zwraca

wskaznik na znaleziony klucz

4.2.1.8 `znaklu()`

```
void znaklu (
    int wysokosc,
    Ekwipunekstr ** pGlowa,
    unsigned char * znak )
```

Funkcja dodaje znaleziony klucz do ekwipunku oraz wypisuje ponownie ekwipunek

Parametry

<i>wysokosc</i>	ilosc wierszy mapy
<i>pGlowa</i>	wskaznik na liste kluczy ktore posiada gracz
<i>znak</i>	jaki klucz zostal znaleziony

4.3 Dokumentacja pliku Header.h

```
#include <stdbool.h>
```

Struktury danych

- struct `pole`
- struct `kordy`
- struct `Ekwipunekstr`
- struct `wrog`
- struct `tabwyn`

Definicje typów

- typedef `pole` **** `wskpole`
- typedef struct `Ekwipunekstr` `Ekwipunekstr`
- typedef struct `wrog` `wrog`
- typedef struct `tabwyn` `tabwyn`

Wyliczenia

- enum `rodzaj` {
 `glr` = 201, `gpr` = 187, `dlr` = 200, `dpr` = 188,
 `poz` = 205, `pio` = 186, `lsc` = 204, `psc` = 185,
 `gsc` = 203, `dsc` = 202, `sro` = 204, `kro` = 248,
 `pus` = 32, `klu1` = 'a', `klu2` = 'b', `klu3` = 'c',
 `klu4`, `klu5`, `klu6`, `zam1` = 'A',
 `zam2` = 'B', `zam3` = 'C', `zam4`, `zam5`,
 `zam6`, `wyjście` = 'X' }
- enum `czmwej` { `tak` = 0, `nie` = 1, `drz` = 2, `kon` = 3 }
- enum `sterowanie` {
 `zkps` = 224, `zkps0` = 0, `lewo` = 75, `prawo` = 77,
 `gora` = 72, `dol` = 80 }
- enum `postacie` { `gracz` = 208, `potwor` = 245 }
- enum `elementydlagracza` {
 `chodzenie` = 1, `eglr`, `egpr`, `edlr`,
 `edpr`, `epoz`, `epio`, `elsc`,
 `epsc`, `egsc`, `edsc`, `esro`,
 `ekro`, `epus`, `eklu1`, `eklu2`,
 `eklu3`, `eklu4`, `eklu5`, `eklu6`,
 `ezam1`, `ezam2`, `ezam3`, `ezam4`,
 `ezam5`, `ezam6`, `ewyjście`, `epotwor`,
 `egracz`, `wyjściezapisem`, `wyjściebezzapisu` }

Funkcje

- void `gotoxy` (int x, int y)
- void `czyszczenie` ()
- void `czyszczeniebufora` ()
- void `podmien` (int wiersz, int kolumna, int znak, int wysokosc)
- void `dowolnyprzycisk` ()

4.3.1 Dokumentacja funkcji

4.3.1.1 czyszczenie()

```
void czyszczenie ( )
```

Funkcja czysci konsrole

4.3.1.2 czyszczeniebufora()

```
void czyszczeniebufora ( )
```

Funkcja usuwa wszystkie elementy z bufora stdin

4.3.1.3 dowolnyprzycisk()

```
void dowolnyprzycisk ( )
```

Funkcja czeka, az gracz wcisnie dowolny klawisz

4.3.1.4 gotoxy()

```
void gotoxy (
    int x,
    int y )
```

Funkcja przemieszcza kursor wpisywania na podana pozycje

Parametry

<i>x</i>	pozycja w pionie
<i>y</i>	pozycja w poziomie

4.3.1.5 podmien()

```
void podmien (
    int wiersz,
    int kolumna,
    int znak,
    int wysokosc )
```

Funkcja wpisuje znak w podanym miejscu

Parametry

<i>wiersz</i>	w którym wierszu ma być wstawiony znak
<i>kolumna</i>	w której kolumnie ma być wstawiony znak
<i>znak</i>	jaki znak ma być wstawiony
<i>wysokosc</i>	mapy

4.4 Dokumentacja pliku obsługi pliku.h

```
#include "Header.h"
```

Funkcje

- void `zczytajtablice` (`tabwyn **pHead`)
- bool `zczytaj` (`wskpole` tablica, `wrog **przeciwnicy`, `int *wysokosc`, `int *szerokosc`, `kordy **postac`, `char *nazwa`)
- bool `czytaniemapy` (`wskpole` tab, `wrog **przeciwnicy`, `int *wysokosc`, `int *szerokosc`, `kordy **postac`)
- void `zapiszmape` (`wskpole` mapa, `wrog **przeciwnicy`, `int wiersze`, `int kolumny`, `kordy *postac`)

4.4.1 Dokumentacja funkcji

4.4.1.1 czytaniemapy()

```
bool czytaniemapy (  
    wskpole tab,  
    wrog ** przeciwnicy,  
    int * wysokosc,  
    int * szerokosc,  
    kordy ** postac )
```

Funkcja obsługuje menu wpisywania nazwy mapy

Parametry

<i>tablica</i>	wskaznik na mape
<i>przeciwnicy</i>	wskaznik na liste przeciwnikow
<i>wysokosc</i>	ilosc wierszy mapy
<i>szerokosc</i>	ilosc kolumn mapy
<i>postac</i>	wskaznik na pozycje postaci

Zwraca

czy gracz chce przerwać wpisywanie mapy i wrócić do menu głównego

4.4.1.2 zapiszmape()

```
void zapiszmape (
    wskpole mapa,
    wrog ** przeciwnicy,
    int wiersze,
    int kolumny,
    kordy * postac )
```

Funkcja zapisuje do pliku mapę

Parametry

<i>tablica</i>	wskaznik na mapę
<i>przeciwnicy</i>	wskaznik na liście przeciwników
<i>wiersze</i>	ilość wierszy mapy
<i>kolumny</i>	ilość kolumn mapy
<i>postac</i>	wskaznik na pozycję postaci

4.4.1.3 zczytaj()

```
bool zczytaj (
    wskpole tablica,
    wrog ** przeciwnicy,
    int * wysokosc,
    int * szerokosc,
    kordy ** postac,
    char * nazwa )
```

Funkcja czytuje plik zawierający mapę

Parametry

<i>tablica</i>	wskaznik na mapę
<i>przeciwnicy</i>	wskaznik na liście przeciwników
<i>wysokosc</i>	ilość wierszy mapy
<i>szerokosc</i>	ilość kolumn mapy
<i>postac</i>	wskaznik na pozycję postaci
<i>nazwa</i>	nazwa mapy

Zwraca

czy otwarcie pliku sie udalo i czy dane sa poptawne

4.4.1.4 zczytajtablice()

```
void zczytajtablice (
    tabwyn ** pHead )
```

Funkcja zczytuje z pliku tablice wynikow

Parametry

<i>pHead</i>	wskaznik na tablice wynikow
--------------	-----------------------------

4.5 Dokumentacja pliku przeciwnicy.h

```
#include "Header.h"
```

Funkcje

- void **dodprzec** (**wrog** **przeciwnicy, **kordy** pozycja, int kierunek, int naczym)
- void **zmianapozycji**(**kordy** *ja, **kordy** *kolja, **pole** ***tablica, int wiersze, unsigned char *tymczasowy)
- **wrog** * **znajdzprzeciwnika** (**kordy** gdzie, **wrog** *przec)
- void **ruchnormalny** (**wrog** *przeciw, **kordy** *kolejne)
- void **ruchodwrotny** (**wrog** *przeciw, **kordy** *kolejne)
- bool **ruchnormalnyzw** (**wrog** *przeciw, **pole** ***tablica, int wysokosc, int szerokosc, bool *koniec, **kordy** *kolejne, **wrog** *wszyscy)
- bool **ruchodwrotnyzw** (**wrog** *przeciw, **pole** ***tablica, int wysokosc, int szerokosc, bool *koniec, **kordy** *kolejne, **wrog** *wszyscy)
- bool **potworprzed** (**wrog** *przeciw, **pole** ***tablica, int wysokosc, int szerokosc, bool *koniec, **kordy** odkogo, bool sciana, **wrog** *wszyscy)
- void **gdzieiscprze** (**wrog** *przeciw, **pole** ***tablica, int wysokosc, int szerokosc, bool *koniec, **wrog** *wszyscy)
- void **wyszczscruch** (**wrog** **przeciw)
- bool **poruszprzeciwnikami** (**wrog** **przeciw, **pole** ***tablica, int wiersze, int kolumny)
- void **usunprzeciw** (**wrog** **przeciwnicy, **kordy** pozycja)

4.5.1 Dokumentacja funkcji

4.5.1.1 dodprzec()

```
void dodprzec (
    wrog ** przeciwnicy,
    kordy pozycja,
    int kierunek,
    int naszym )
```

Funkcja dodaje przeciwnika do listy przeciwnikow

Parametry

<i>przeciwnicy</i>	lista przeciwnikow
<i>pozycja</i>	miejsce w ktorym znajduje sie przeciwnik
<i>kierunek</i>	kierunek, w ktorym porusz sie przeciwnik
<i>naczym</i>	na jakim polu stoi przeciwnik

4.5.1.2 gdzieiscprze()

```
void gdzieiscprze (
    wrog * przeciwi,
    pole *** tablica,
    int wysokosc,
    int szerokosc,
    bool * koniec,
    wrog * wszyscy )
```

Funkcja wykonuje ruch jednego przeciwnika

Parametry

<i>przeciwi</i>	aktualnie ruszajacy sie przeciwnik
<i>tablica</i>	wkaznik na mape
<i>wysokosc</i>	ilosc wierszy mapy
<i>szerokosc</i>	ilosc kolumn mapy
<i>koniec</i>	czy gracz zostal zabity
<i>wszyscy</i>	wskaznika na liste przeciwnikow

4.5.1.3 poruszprzeciwnikami()

```
bool poruszprzeciwnikami (
    wrog ** przeciwi,
    pole *** tablica,
    int wiersze,
    int kolumny )
```

Funkcja wykonuje ruchy wszystkich przeciwnikow

Parametry

<i>przeciwi</i>	wskaznik na liste przeciwnikow
<i>tablica</i>	wskaznik na mape
<i>wiersze</i>	ilosc wierszy mapy
<i>kolumny</i>	ilosc kolumn mapy

Zwraca

czy gracz zostal zabity

4.5.1.4 potworprzed()

```
bool potworprzed (
    wrog * przeciwiw,
    pole *** tablica,
    int wysokosc,
    int szerokosc,
    bool * koniec,
    kordy odkogo,
    bool sciana,
    wrog * wszyscy )
```

Funkcja obsluguje sytuacje gdy na drodze przeciwnika pojawia sie drugi przeciwnik

Parametry

<i>przeciwiw</i>	aktualnie ruszajacy sie przeciwnik
<i>tablica</i>	wkaznik na mape
<i>wysokosc</i>	ilosc wierszy mapy
<i>szerokosc</i>	ilosc kolumn mapy
<i>koniec</i>	czy gracz zostal zabity
<i>odkogo</i>	ktory przeciwnik kazal ruszyc sie aktualnemu przeciwnikowi
<i>sciana</i>	czy przeciwnik wykonuje ruch odwrotny
<i>wszyscy</i>	wskaznika na liste przeciwnikow

Zwraca

czy udalo sie wykonac ruch

4.5.1.5 ruchnormalny()

```
void ruchnormalny (
    wrog * przeciwiw,
    kordy * kolejne )
```

Funkcja podaje nastepna pozycje przeciwnika

Parametry

<i>przeciwiw</i>	aktualnie ruszajacy sie przeciwnik
<i>kolejne</i>	kolejna pozycja przeciwnika

4.5.1.6 ruchnormalnyzw()

```
bool ruchnormalnyzw (
    wrog * przeciwiw,
    pole *** tablica,
    int wysokosc,
    int szerokosc,
    bool * koniec,
    kordy * kolejne,
    wrog * wszyscy )
```

Funkcja obsługuje ruch, gdy potwór idzie zgodnie z aktualnym kierunkiem ruchu

Parametry

<i>przeciwiw</i>	aktualnie ruszający się przeciwnik
<i>tablica</i>	wkaznik na mapę
<i>wysokosc</i>	ilość wierszy mapy
<i>szerokosc</i>	ilość kolumn mapy
<i>koniec</i>	czy gracz został zabity
<i>kolejne</i>	kolejna pozycja przeciwnika
<i>wszyscy</i>	wskaznik na listę przeciwników

Zwraca

czy udało się wykonać ruch

4.5.1.7 ruchodwrotny()

```
void ruchodwrotny (
    wrog * przeciwiw,
    kordy * kolejne )
```

Funkcja podaje pozycję pozycji przeciwnika pod zmianę kierunku oraz zmienia kierunek poruszania się przeciwnika

Parametry

<i>przeciwiw</i>	aktualnie ruszający się przeciwnik
<i>kolejne</i>	kolejna pozycja przeciwnika

4.5.1.8 ruchodwrotnyzw()

```
bool ruchodwrotnyzw (
```

```
wrog * przeciwiw,  
pole *** tablica,  
int wysokosc,  
int szerokosc,  
bool * koniec,  
kordy * kolejne,  
wrog * wszyscy )
```

Funkcja obsługuje ruch, gdy potwór nie może iść zgodnie z wcześniejszym kierunkiem ruchu

Parametry

<i>przeciwiw</i>	aktualnie ruszający się przeciwnik
<i>tablica</i>	wskaznik na mapę
<i>wysokosc</i>	ilość wierszy mapy
<i>szerokosc</i>	ilość kolumn mapy
<i>koniec</i>	czy gracz został zabity
<i>kolejne</i>	kolejna pozycja przeciwnika
<i>wszyscy</i>	wskaznik na listę przeciwników

Zwraca

czy udało się wykonać ruch

4.5.1.9 usunprzeciwiw()

```
void usunprzeciwiw (  
    wrog ** przeciwnicy,  
    kordy pozycja )
```

Funkcja usuwa przeciwnika znajdującego się na danej pozycji na mapie z listy przeciwników

Parametry

<i>przeciwnicy</i>	wskaznik na listę przeciwników
<i>pozycja</i>	pozycja usuwanego przeciwnika

4.5.1.10 wyszczscruch()

```
void wyszczscruch (  
    wrog ** przeciwiw )
```

Funkcja czyści znacznik wykonania ruchu przez przeciwników w danej turze

Parametry

<i>przeciw</i>	wskaznik na liste przeciwnikow
----------------	--------------------------------

4.5.1.11 zmianapozycjiprze()

```
void zmianapozycjiprze (
    kordy * ja,
    kordy * kolja,
    pole *** tablica,
    int wiersze,
    unsigned char * tymczasowy )
```

Funkcja zmienia pozycje przeciwnika na mapie

Parametry

<i>ja</i>	aktualna pozycja przeciwnika
<i>kolja</i>	nastepna pozycja przeciwnika
<i>tablica</i>	wskaznik na mape
<i>wiersze</i>	ilosc wierszy mapy
<i>tymczasowy</i>	na jakim polu stoi przeciwnik

4.5.1.12 znajdzprzeciwnika()

```
wrog* znajdzprzeciwnika (
    kordy gdzie,
    wrog * przec )
```

Funkcja szuka przeciwnika znajdujacego sie na danej pozycji w liscie przeciwnikow

Parametry

<i>gdzie</i>	pozycja przeciwnika
<i>przec</i>	lista przeciwnikow

Zwraca

wskaznik na znalezionego przeciwnika

4.6 Dokumentacja pliku tablicawynikow.h

```
#include "Header.h"
```

Funkcje

- void `dodajwynik` (`tabwyn **pHead`, `tabwyn *nowy`)
- void `tablicawynikow` (`tabwyn *pHead`)
- void `wipiszdotablicy` (`tabwyn **pHead`, `char nazwa[30]`, `int ptk`)
- void `zapisztablice` (`tabwyn *pHead`)

4.6.1 Dokumentacja funkcji

4.6.1.1 dodajwynik()

```
void dodajwynik (
    tabwyn ** pHead,
    tabwyn * nowy )
```

Funkcja dodaje nowy wynik do listy wynikow

Parametry

<i>pHead</i>	wskaznik na tablice wynikow
<i>nowy</i>	wstawiany element tablicy

4.6.1.2 tablicawynikow()

```
void tablicawynikow (
    tabwyn * pHead )
```

Funkcja wypisuje tablice wynikow

Parametry

<i>pHead</i>	wskaznik na tablice wynikow
--------------	-----------------------------

4.6.1.3 wipiszdotablicy()

```
void wipiszdotablicy (
    tabwyn ** pHead,
    char nazwa[30],
    int ptk )
```

Funkcja wpisuje do tablicy uzyskany przez gracza wynik

Parametry

<i>pHead</i>	wskaznik na tablice wynikow
<i>nazwa</i>	imie gracza
<i>ptk</i>	ilosc punktow jakie uzyskal gracz

4.6.1.4 zapisztabelle()

```
void zapisztabelle (
    tabwyn * pHead )
```

Funkcja zapisuje tablice wynikow do pliku

Parametry

<i>pHead</i>	wskaznik na tablice wynikow
--------------	-----------------------------

4.7 Dokumentacja pliku usuwanie.h

```
#include "Header.h"
```

Funkcje

- void `usuntabelle` (`tabwyn **lista`)
- void `usunmape` (`wskpole` mapa, int wiersze, int kolumny)
- void `usunliste` (`wrog **przeciwnik`)
- void `usuncalamape` (`wskpole` mapa, int wiersze, int kolumny, `wrog **przeciwnik`, `kordy **postac`)
- void `usunekwipunek` (`Ekwipunekstr **pHeadekw`)
- void `usunnazwe` (`char **nazwa`)

4.7.1 Dokumentacja funkcji

4.7.1.1 usuncalamape()

```
void usuncalamape (
    wskpole mapa,
    int wiersze,
    int kolumny,
    wrog ** przeciwnik,
    kordy ** postac )
```

Funkcja zwalnia pamiec zajeta przez mape oraz liste przeciwnikow i postac

Parametry

<i>mapa</i>	wskaznik na mape
<i>wiersze</i>	ilosc wierszy mapy
<i>kolumny</i>	ilisc kolumn mapy
<i>przeciwnik</i>	wskaznik na liste przeciwnikow
<i>postac</i>	wskaznik na miejsce, w ktorym znajduje sie postac

4.7.1.2 usunekwipunek()

```
void usunekwipunek (
    Ekwipunekstr ** pHeadekw )
```

Funkcja zwalnia pamiec zajeta przez liste przedmiotow

Parametry

<i>pHeadekw</i>	wskaznik na liste przedmiotow
-----------------	-------------------------------

4.7.1.3 usunliste()

```
void usunliste (
    wrog ** przeciwnik )
```

Funkcja zwalnia pamiec zajenta przez liste przeciwnikow

Parametry

<i>przeciwnik</i>	wskaznik na liste przeciwnikow
-------------------	--------------------------------

4.7.1.4 usunmape()

```
void usunmape (
    wskpole mapa,
    int wiersze,
    int kolumny )
```

Funkcja zwalnia pamiec zajeta przez mape

Parametry

<i>mapa</i>	wskaznik na mape
<i>wiersze</i>	ilosc wierszy mapy
<i>kolumny</i>	ilisc kolumn mapy

4.7.1.5 usunnazwe()

```
void usunnazwe (
    char ** nazwa )
```

Funkcja zwalnia pamiec zajenta przez nazwe gracza

Parametry

<i>nazwa</i>	imie uzytkownika
--------------	------------------

4.7.1.6 usuntabllice()

```
void usuntabllice (
    tabwyn ** lista )
```

Funkcja zwalnia pamiec zajenta przez liste wynikow

Parametry

<i>lista</i>	wskaznik na liste wynikow
--------------	---------------------------

4.8 Dokumentacja pliku wypisywanie.h

```
#include "Header.h"
```

Funkcje

- void [wypiszekw](#) ([Ekwipunekstr](#) *pGlowa)
- void [wypiszmapiekw](#) ([pole](#) ***tablica, int wiersze, int kolumny, [Ekwipunekstr](#) *ekw)
- void [wypiszmapa](#) ([pole](#) ***tablica, int wiersze, int kolumny)
- void [wypiszpomoc](#) (int szerokosc)
- void [wypiszpomchodzenie](#) (int kolumny)

4.8.1 Dokumentacja funkcji

4.8.1.1 wypiszekw()

```
void wypiszekw (
    Ekwipunekstr * pGlowa )
```

Funkcja wypisuje w konsoli ekwipunek

Parametry

<i>pGlowa</i>	wskaznik na liste przedmiotow
---------------	-------------------------------

4.8.1.2 wypiszmapa()

```
void wypiszmapa (
    pole *** tablica,
    int wiersze,
    int kolumny )
```

Funkcja wypisuje w konsoli mape

Parametry

<i>tablica</i>	wskaznik na mape
<i>wiersze</i>	ilosc wierszy mapy
<i>kolumny</i>	ilosc kolumn mapy

4.8.1.3 wypiszmapiekw()

```
void wypiszmapiekw (
    pole *** tablica,
    int wiersze,
    int kolumny,
    Ekwipunekstr * ekw )
```

Funkcja wypisuje w konsoli mape i ekwipunek

Parametry

<i>tablica</i>	wskaznik na mape
<i>wiersze</i>	ilosc wierszy mapy
<i>kolumny</i>	ilosc kolumn mapy
<i>ekw</i>	wskaznik na liste przedmiotow

4.8.1.4 wypiszpomchodzenie()

```
void wypiszpomchodzenie (
    int kolumny )
```

Funkcja wypisuje pomoc w trybie chodzenia w edytorze map

Parametry

<i>kolumny</i>	ilosc kolumn mapy
----------------	-------------------

4.8.1.5 wypiszpomoc()

```
void wypiszpomoc (  
    int szerokosc )
```

Funkcja wypisuje pomoc w edytorze map

Parametry

<i>szerokosc</i>	ilosc kolumn mapy
------------------	-------------------

Indeks

- czyszczenie
 - Header.h, 15
- czyszczeniebufora
 - Header.h, 15
- czytaniemap
 - obsługapliku.h, 16
- dodajwynik
 - tablicawynikow.h, 25
- dodlista
 - gra.h, 9
- dodprzec
 - przeciwnicy.h, 18
- dowolnyprzycisk
 - Header.h, 15
- edycja
 - edytormap.h, 7
- edytormap
 - edytormap.h, 8
- edytormap.h, 7
 - edycja, 7
 - edytormap, 8
 - nowamapa, 8
 - przemieszczenie, 8
 - staramapa, 8
 - wktoastrone, 8
 - wpisz, 8
- Ekwipunekstr, 5
- gdzieiscprze
 - przeciwnicy.h, 20
- gotoxy
 - Header.h, 15
- gra
 - gra.h, 10
- gra.h, 9
 - dodlista, 9
 - gra, 10
 - klawiatura, 10
 - otwdrzwi, 10
 - poruszanie, 12
 - ruch, 12
 - znajdzwekw, 13
 - znaklu, 13
- Header.h, 14
 - czyszczenie, 15
 - czyszczeniebufora, 15
 - dowolnyprzycisk, 15
 - gotoxy, 15
 - podmien, 15
- klawiatura
 - gra.h, 10
- kordy, 5
- nowamapa
 - edytormap.h, 8
- obsługapliku.h, 16
 - czytaniemap, 16
 - zapiszmap, 17
 - zczytaj, 17
 - zczytajtablice, 18
- otwdrzwi
 - gra.h, 10
- podmien
 - Header.h, 15
- pole, 6
- poruszanie
 - gra.h, 12
- poruszprzeciwnikami
 - przeciwnicy.h, 20
- potworprzed
 - przeciwnicy.h, 21
- przeciwnicy.h, 18
 - dodprzec, 18
 - gdzieiscprze, 20
 - poruszprzeciwnikami, 20
 - potworprzed, 21
 - ruchnormalny, 21
 - ruchnormalnyzw, 22
 - ruchodwrotny, 22
 - ruchodwrotnyzw, 22
 - usunprzeciw, 23
 - wyszczscruch, 23
 - zmianapozycjiprze, 24
 - znajdzprzeciwnika, 24
- przemieszczenie
 - edytormap.h, 8
- ruch
 - gra.h, 12
- ruchnormalny
 - przeciwnicy.h, 21
- ruchnormalnyzw
 - przeciwnicy.h, 22
- ruchodwrotny
 - przeciwnicy.h, 22

ruchodwrotnyzw
przeciwnicy.h, 22

staramapa
edytormap.h, 8

tablicawynikow
tablicawynikow.h, 25

tablicawynikow.h, 24
dodajwynik, 25
tablicawynikow, 25
wipiszdotablicy, 25
zapisztabelle, 26

tabwyn, 6

usuncalamape
usuwanie.h, 26

usunekwipunek
usuwanie.h, 27

usunliste
usuwanie.h, 27

usunmape
usuwanie.h, 27

usunnazwe
usuwanie.h, 28

usunprzeciw
przeciwnicy.h, 23

usuntabelle
usuwanie.h, 28

usuwanie.h, 26
usuncalamape, 26
usunekwipunek, 27
usunliste, 27
usunmape, 27
usunnazwe, 28
usuntabelle, 28

wipiszdotablicy
tablicawynikow.h, 25

wktorastrone
edytormap.h, 8

wpisz
edytormap.h, 8

wrog, 6

wypisywanie.h, 28
wypiszekw, 28
wypiszmapa, 29
wypiszmapiekw, 29
wypiszpomchodzenie, 29
wypiszpomoc, 30

wypiszekw
wypisywanie.h, 28

wypiszmapa
wypisywanie.h, 29

wypiszmapiekw
wypisywanie.h, 29

wypiszpomchodzenie
wypisywanie.h, 29

wypiszpomoc
wypisywanie.h, 30

wyszczscruch
przeciwnicy.h, 23

zapiszmape
obslugapliku.h, 17

zapisztabelle
tablicawynikow.h, 26

zczytaj
obslugapliku.h, 17

zczytajtabelle
obslugapliku.h, 18

zmianapozycjiprze
przeciwnicy.h, 24

znajdzprzeciwnika
przeciwnicy.h, 24

znajdzwekw
gra.h, 13

znaku
gra.h, 13