

Koło Fortuny

Generated by Doxygen 1.8.17

1 File Index	1
1.1 File List	1
2 File Documentation	3
2.1 funkcje.c File Reference	3
2.1.1 Function Documentation	3
2.1.1.1 czytajZPliku()	3
2.1.1.2 koloFortuny()	4
2.1.1.3 liczWystapieniaLiteryWHasle()	4
2.1.1.4 obslugaArgumentow()	4
2.1.1.5 obslugaZnaku()	5
2.1.1.6 sprawdzCzyWygrana()	5
2.1.1.7 wprowadzanieZnaku()	5
2.1.1.8 wygrana()	6
2.1.1.9 wypiszAktualnyStanTablicy()	6
2.1.1.10 wypiszZasady()	6
2.1.1.11 zakryjHaslo()	6
2.1.1.12 zgadywanieHasla()	7
2.1.1.13 znakJestLitera()	7
2.2 funkcje.h File Reference	7
2.2.1 Enumeration Type Documentation	8
2.2.1.1 bool	8
2.2.2 Function Documentation	8
2.2.2.1 czytajZPliku()	8
2.2.2.2 koloFortuny()	8
2.2.2.3 liczWystapieniaLiteryWHasle()	9
2.2.2.4 obslugaArgumentow()	9
2.2.2.5 obslugaZnaku()	9
2.2.2.6 sprawdzCzyWygrana()	9
2.2.2.7 wprowadzanieZnaku()	10
2.2.2.8 wygrana()	10
2.2.2.9 wypiszAktualnyStanTablicy()	10
2.2.2.10 wypiszZasady()	11
2.2.2.11 zakryjHaslo()	11
2.2.2.12 zgadywanieHasla()	11
2.2.2.13 znakJestLitera()	11
Index	13

Chapter 1

File Index

1.1 File List

Here is a list of all documented files with brief descriptions:

funkcje.c	3
funkcje.h	7

Chapter 2

File Documentation

2.1 funkcje.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include "funkcje.h"
```

Functions

- void **wypiszZasady** ()
- void **obsługaArgumentow** ()
- char * **czytajZPliku** (char nazwaPliku[])
- char * **zakryjHaslo** (char *haslo)
- void **wypiszAktualnyStanTablicy** (char litera, char aktualnyStanTablicy[], char haslo[])
- char **wprowadzanieZnaku** (char wprowadzonyZnak)
- void **wygrana** (int *stanKonta)
- int **liczWystapieniaLiteryWHasle** (char wprowadzonaLitera, char haslo[], char aktualnyStanTablicy[])
- int **koloFortuny** (int wystapieniaLiteryWHasle, int *stanKonta)
- bool **sprawdzCzyWygrana** (char haslo[], char aktualnyStanTablicy[])
- bool **znakJestLitera** (char wprowadzonaLitera, char haslo[], char aktualnyStanTablicy[], int *stanKonta)
- bool **zgadywanieHasla** (char haslo[])
- bool **obsługaZnaku** (char wprowadzonyZnak, char haslo[], char aktualnyStanTablicy[], int *stanKonta)

2.1.1 Function Documentation

2.1.1.1 czytajZPliku()

```
char* czytajZPliku (
    char nazwaPliku[] )
```

Losowanie i odczytywanie z pliku hasła.

Parameters

<i>nazwaPliku[]</i>	nazwa pliku z hasłami
---------------------	-----------------------

2.1.1.2 koloFortuny()

```
int koloFortuny (
    int wystapieniaLitereyWHasle,
    int * stanKonta )
```

Losowanie wartości od 0 do 10 będących ilością punktów za wprowadzoną spółgłoskę. Wylosowanie 0 oznacza wyzerowanie konta.

Parameters

<i>wystapieniaLitereyWHasle</i>	liczba wystąpień litery we wzorze hasła
<i>stanKonta</i>	aktualny stan konta

2.1.1.3 liczWystapieniaLitereyWHasle()

```
int liczWystapieniaLitereyWHasle (
    char wprowadzonaLitera,
    char haslo[],
    char aktualnyStanTablicy[] )
```

Zliczanie wystąpień litery w hasle mnożnika punktów w wypadku wprowadzenia spółgłoski.

Parameters

<i>wprowadzonaLitera</i>	wprowadzony przez użytkownika znak rozpoznany jako litera
<i>haslo</i>	wzór hasła
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter

2.1.1.4 obslugaArgumentow()

```
void obslugaArgumentow ( )
```

Wyświetlanie w oknie konsoli instrukcji obsługi parametrów.

2.1.1.5 obslugaZnaku()

```
bool obslugaZnaku (
    char wprowadzonyZnak,
    char haslo[],
    char aktualnyStanTablicy[],
    int * stanKonta )
```

Definiowanie czy użytkownik wprowadził poprawny znak będący literą lub czy chce podać pełne hasło.

Parameters

<i>wprowadzonyZnak</i>	wprowadzony przez użytkownika znak jeszcze niezdefiniowany jako litera, bądź "?"
<i>haslo</i>	wzór hasła
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter
<i>stanKonta</i>	aktualny stan konta

2.1.1.6 sprawdzCzyWygrana()

```
bool sprawdzCzyWygrana (
    char haslo[],
    char aktualnyStanTablicy[] )
```

Sprawdzenie czy aktualny stan tablicy jest taki sam jak wzor hasla.

Parameters

<i>haslo</i>	wzór hasła
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter

2.1.1.7 wprowadzanieZnaku()

```
char wprowadzanieZnaku (
    char wprowadzonyZnak )
```

Obsługa wprowadzania pojedynczej litery.

Parameters

<i>wprowadzonyZnak</i>	wprowadzony przez użytkownika znak jeszcze niezdefiniowany jako litera, bądź "?"
------------------------	--

2.1.1.8 wygrana()

```
void wygrana (
    int * stanKonta )
```

Powiadomienie użytkownika o wygranej rundzie i wyświetlenie stanu konta.

Parameters

<i>stanKonta</i>	aktualny stan konta
------------------	---------------------

2.1.1.9 wypiszAktualnyStanTablicy()

```
void wypiszAktualnyStanTablicy (
    char litera,
    char aktualnyStanTablicy[],
    char haslo[] )
```

Wypisywanie w oknie konsoli aktualnego stanu odkrytych i nieodkrytych liter hasła.

Parameters

<i>litera</i>	wprowadzona przez użytkownika litera
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter
<i>haslo</i>	wzór hasła

2.1.1.10 wypiszZasady()

```
void wypiszZasady ( )
```

Wyświetlanie zasad gry, jeżeli jako parametr podano '-z'.

2.1.1.11 zakryjHaslo()

```
char* zakryjHaslo (
    char * haslo )
```

Zakrywanie hasła w oknie konsoli użytkownik będzie widzieć znaki w ilości i konfiguracji odpowiadającej tej we wzorze hasła.

Parameters

<i>haslo</i>	wzór hasła
--------------	------------

2.1.1.12 zgadywanieHasla()

```
bool zgadywanieHasla (
    char haslo[] )
```

Sprawdzenie czy ciąg znaków podanych po wprowadzeniu przez użytkownika "?" jest zgodny ze wzorem hasła.

Parameters

<i>haslo</i>	wzór hasła
--------------	------------

2.1.1.13 znakJestLitera()

```
bool znakJestLitera (
    char wprowadzonaLitera,
    char haslo[],
    char aktualnyStanTablicy[],
    int * stanKonta )
```

Decyzja co zrobić dalej, gdy już wiadomo, że wprowadzony znak jest literą.

Parameters

<i>wprowadzonaLitera</i>	wprowadzony przez użytkownika znak rozpoznany jako litera
<i>haslo</i>	wzór hasła
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter
<i>stanKonta</i>	aktualny stan konta

2.2 funkcje.h File Reference

Enumerations

- enum **bool** { **false**, **true** }

Functions

- void **obsługaArgumentow** ()
- void **wypiszZasady** ()
- char * **czytajZPliku** (char nazwaPliku[])
- char * **zakryjHaslo** (char *haslo)
- void **wypiszAktualnyStanTablicy** (char litera, char aktualnyStanTablicy[], char haslo[])
- char **wprowadzanieZnaku** (char wprowadzonyZnak)

- void **wygrana** (int *stanKonta)
- int **liczWystapieniaLiteryWHasle** (char wprowadzonaLitera, char haslo[], char aktualnyStanTablicy[])
- int **koloFortuny** (int wystapieniaLiteryWHasle, int *stanKonta)
- bool **sprawdzCzyWygrana** (char haslo[], char aktualnyStanTablicy[])
- bool **znakJestLitera** (char wprowadzonaLitera, char haslo[], char aktualnyStanTablicy[], int *stanKonta)
- bool **zgadywanieHasla** (char haslo[])
- bool **obsługaZnaku** (char wprowadzonyZnak, char haslo[], char aktualnyStanTablicy[], int *stanKonta)

2.2.1 Enumeration Type Documentation

2.2.1.1 bool

```
enum bool
```

Dla uproszczenia utworzony został typ bool.

2.2.2 Function Documentation

2.2.2.1 czytajZPliku()

```
char* czytajZPliku (
    char nazwaPliku[] )
```

Losowanie i odczytywanie z pliku hasła.

Parameters

<i>nazwaPliku[]</i>	nazwa pliku z hasłami
---------------------	-----------------------

2.2.2.2 koloFortuny()

```
int koloFortuny (
    int wystapieniaLiteryWHasle,
    int * stanKonta )
```

Losowanie wartości od 0 do 10 będących ilością punktów za wprowadzoną spółgłoskę. Wylosowanie 0 oznacza wyzerowanie konta.

Parameters

<i>wystapieniaLiteryWHasle</i>	liczba wystąpień litery we wzorze hasła
<i>stanKonta</i>	aktualny stan konta

2.2.2.3 liczWystapieniaLitereWHasle()

```
int liczWystapieniaLitereWHasle (
    char wprowadzonaLitera,
    char haslo[],
    char aktualnyStanTablicy[] )
```

Zliczanie wystąpień litery w hasle mnożnika punktów w wypadku wprowadzenia spółgłoski.

Parameters

<i>wprowadzonaLitera</i>	wprowadzony przez użytkownika znak rozpoznany jako litera
<i>haslo</i>	wzór hasła
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter

2.2.2.4 obslugaArgumentow()

```
void obslugaArgumentow ( )
```

Wyświetlanie w oknie konsoli instrukcji obsługi parametrów.

2.2.2.5 obslugaZnaku()

```
bool obslugaZnaku (
    char wprowadzonyZnak,
    char haslo[],
    char aktualnyStanTablicy[],
    int * stanKonta )
```

Definiowanie czy użytkownik wprowadził poprawny znak będący literą lub czy chce podać pełne hasło.

Parameters

<i>wprowadzonyZnak</i>	wprowadzony przez użytkownika znak jeszcze niezdefiniowany jako litera, bądź "?"
<i>haslo</i>	wzór hasła
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter
<i>stanKonta</i>	aktualny stan konta

2.2.2.6 sprawdzCzyWygrana()

```
bool sprawdzCzyWygrana (
    char haslo[],
    char aktualnyStanTablicy[] )
```

Sprawdzenie czy aktualny stan tablicy jest taki sam jak wzor hasla.

Parameters

<i>haslo</i>	wzór hasła
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter

2.2.2.7 wprowadzanieZnaku()

```
char wprowadzanieZnaku (
    char wprowadzonyZnak )
```

Obsługa wprowadzania pojedynczej litery.

Parameters

<i>wprowadzonyZnak</i>	wprowadzony przez użytkownika znak jeszcze niezdefiniowany jako litera, bądź "?"
------------------------	--

2.2.2.8 wygrana()

```
void wygrana (
    int * stanKonta )
```

Powiadomienie użytkownika o wygranej rundzie i wyświetlenie stanu konta.

Parameters

<i>stanKonta</i>	aktualny stan konta
------------------	---------------------

2.2.2.9 wypiszAktualnyStanTablicy()

```
void wypiszAktualnyStanTablicy (
    char litera,
    char aktualnyStanTablicy[],
    char haslo[] )
```

Wypisywanie w oknie konsoli aktualnego stanu odkrytych i nieodkrytych liter hasła.

Parameters

<i>litera</i>	wprowadzona przez użytkownika litera
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter
<i>haslo</i>	wzór hasła

2.2.2.10 wypiszZasady()

```
void wypiszZasady ( )
```

Wyświetlanie zasad gry, jeżeli jako parametr podano '-z'.

2.2.2.11 zakryjHaslo()

```
char* zakryjHaslo (
    char * haslo )
```

Zakrywanie hasła w oknie konsoli użytkownik będzie widzieć znaki w ilości i konfiguracji odpowiadającej tej we wzorze hasła.

Parameters

<i>haslo</i>	wzór hasła
--------------	------------

2.2.2.12 zgadywanieHasla()

```
bool zgadywanieHasla (
    char haslo[] )
```

Sprawdzenie czy ciąg znaków podanych po wprowadzeniu przez użytkownika "?" jest zgodny ze wzorem hasła.

Parameters

<i>haslo</i>	wzór hasła
--------------	------------

2.2.2.13 znakJestLitera()

```
bool znakJestLitera (
    char wprowadzonaLitera,
    char haslo[],
    char aktualnyStanTablicy[],
    int * stanKonta )
```

Decyzja co zrobić dalej, gdy już wiadomo, że wprowadzony znak jest literą.

Parameters

<i>wprowadzonaLitera</i>	wprowadzony przez użytkownika znak rozpoznany jako litera
<i>haslo</i>	wzór hasła
<i>aktualnyStanTablicy[]</i>	tablica z ciągiem odkrytych i nieodkrytych liter
<i>stanKonta</i>	aktualny stan konta

Index

- bool
 - funkcje.h, 8
- czytajZPliku
 - funkcje.c, 3
 - funkcje.h, 8
- funkcje.c, 3
 - czytajZPliku, 3
 - kołoFortuny, 4
 - liczWystapieniaLiteryWHasle, 4
 - obsługaArgumentów, 4
 - obsługaZnaku, 4
 - sprawdźCzyWygrana, 5
 - wprowadzanieZnaku, 5
 - wygrana, 5
 - wypiszAktualnyStanTablicy, 6
 - wypiszZasady, 6
 - zakryjHasło, 6
 - zgadywanieHasła, 7
 - znakJestLitera, 7
- funkcje.h, 7
 - bool, 8
 - czytajZPliku, 8
 - kołoFortuny, 8
 - liczWystapieniaLiteryWHasle, 9
 - obsługaArgumentów, 9
 - obsługaZnaku, 9
 - sprawdźCzyWygrana, 9
 - wprowadzanieZnaku, 10
 - wygrana, 10
 - wypiszAktualnyStanTablicy, 10
 - wypiszZasady, 11
 - zakryjHasło, 11
 - zgadywanieHasła, 11
 - znakJestLitera, 11
- kołoFortuny
 - funkcje.c, 4
 - funkcje.h, 8
- liczWystapieniaLiteryWHasle
 - funkcje.c, 4
 - funkcje.h, 9
- obsługaArgumentów
 - funkcje.c, 4
 - funkcje.h, 9
- obsługaZnaku
 - funkcje.c, 4
 - funkcje.h, 9
- sprawdźCzyWygrana
 - funkcje.c, 5
 - funkcje.h, 9
- wprowadzanieZnaku
 - funkcje.c, 5
 - funkcje.h, 10
- wygrana
 - funkcje.c, 5
 - funkcje.h, 10
- wypiszAktualnyStanTablicy
 - funkcje.c, 6
 - funkcje.h, 10
- wypiszZasady
 - funkcje.c, 6
 - funkcje.h, 11
- zakryjHasło
 - funkcje.c, 6
 - funkcje.h, 11
- zgadywanieHasła
 - funkcje.c, 7
 - funkcje.h, 11
- znakJestLitera
 - funkcje.c, 7
 - funkcje.h, 11