Koło Fortuny

Generated by Doxygen 1.8.17

Chapter 1

File Index

1.1 File List

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

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 **obslugaArgumentow** ()
- 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** **obslugaZnaku** (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

nazwaPliku[ ]	nazwa pliku z hasłami

2.1.1.2 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

wystapieniaLiteryWHasle	liczba wystąpień litery we wzorze hasła
stanKonta	aktualny stan konta

2.1.1.3 liczWystapieniaLiteryWHasle()

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

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

Parameters

wprowadzonaLitera	wprowadzony przez użytkownika znak rozpoznany jako litera
haslo	wzór hasła
aktualnyStanTablicy[ ]	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

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

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

2.1.1.7 wprowadzanieZnaku()

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

Obsługa wprowadzania pojedynczej litery.

Parameters

wprowadzonyZnak	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

stanKonta	aktualny stan konta

2.1.1.9 wypiszAktualnyStanTablicy()

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

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

Parameters

litera	wprowadzona przez użytkownika litera
aktualnyStanTablicy[]	tablica z ciągiem odkrytych i nieodkrytych liter
haslo	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 hasla w oknie konsoli użytkownik będzie widzieć znaki w ilości i konfiguracji odpowiadającej tej we wzorze hasła.

Parameters

haslo	wzór hasła

2.1.1.12 zgadywanieHasla()

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

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

Parameters

haslo	wzór hasła

2.1.1.13 znakJestLitera()

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

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

Parameters

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

2.2 funkcje.h File Reference

Enumerations

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

Functions

- void **obslugaArgumentow** ()
- 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 obslugaZnaku** (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

nazwaPliku[ ]	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

wystapieniaLiteryWHasle	liczba wystąpień litery we wzorze hasła
stanKonta	aktualny stan konta

2.2.2.3 liczWystapieniaLiteryWHasle()

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

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

Parameters

wprowadzonaLitera	wprowadzony przez użytkownika znak rozpoznany jako litera
haslo	wzór hasła
aktualnyStanTablicy[]	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

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

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

2.2.2.7 wprowadzanieZnaku()

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

Obsługa wprowadzania pojedynczej litery.

Parameters

wprowadzonyZnak	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

stanKonta	aktualny stan konta

2.2.2.9 wypiszAktualnyStanTablicy()

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

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

Parameters

litera	wprowadzona przez użytkownika litera
aktualnyStanTablicy[ ]	tablica z ciągiem odkrytych i nieodkrytych liter
haslo	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 hasla w oknie konsoli użytkownik będzie widzieć znaki w ilości i konfiguracji odpowiadającej tej we wzorze hasła.

Parameters

haslo	wzór hasła

2.2.2.12 zgadywanieHasla()

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

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

Parameters

haslo	wzór hasła

2.2.2.13 znakJestLitera()

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

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

Parameters

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

Index