

Millionairs

Generated by Doxygen 1.8.16

1 Data Structure Index	1
1.1 Data Structures	1
2 File Index	3
2.1 File List	3
3 Data Structure Documentation	5
3.1 leaderboard Struct Reference	5
3.1.1 Detailed Description	5
3.1.2 Field Documentation	5
3.1.2.1 how_much_money	5
3.1.2.2 name	5
3.1.2.3 pNext	6
3.2 question Struct Reference	6
3.2.1 Detailed Description	6
3.2.2 Field Documentation	6
3.2.2.1 answer_A	6
3.2.2.2 answer_B	6
3.2.2.3 answer_C	7
3.2.2.4 answer_D	7
3.2.2.5 correct_answer	7
3.2.2.6 has_it_appeared	7
3.2.2.7 number_of_question	7
3.2.2.8 pNext	7
3.2.2.9 question	7
4 File Documentation	9
4.1 Milionerzy_Projekt_Marek_Kawalski/funkcje.c File Reference	9
4.1.1 Macro Definition Documentation	10
4.1.1.1 _CRT_SECURE_NO_WARNINGS	10
4.1.2 Function Documentation	10
4.1.2.1 about()	10
4.1.2.2 amount_of_questions()	10
4.1.2.3 ask_the_audience()	11
4.1.2.4 call_a_friend()	11
4.1.2.5 check_guard()	11
4.1.2.6 create_a_single_linked_list_by_pushing_back()	12
4.1.2.7 create_list_of_players_by_pushing_front()	13
4.1.2.8 delete_list()	13
4.1.2.9 delete_list_of_players()	13
4.1.2.10 display_name()	14
4.1.2.11 fifty_fifty()	14
4.1.2.12 game_manu()	14

4.1.2.13 generate_random_number()	15
4.1.2.14 give_us_random_question()	15
4.1.2.15 how_much_money()	16
4.1.2.16 how_to_play()	16
4.1.2.17 print_list()	17
4.1.2.18 print_list_of_players()	17
4.1.2.19 read_from_file()	17
4.1.2.20 search_question()	18
4.1.2.21 sub_menu()	18
4.1.2.22 wait()	19
4.1.2.23 what_question()	19
4.1.2.24 write_to_file()	19
4.2 Milionerzy_Projekt_Marek_Kawalski/funkcje.h File Reference	20
4.2.1 Macro Definition Documentation	21
4.2.1.1 _CRT_SECURE_NO_WARNINGS	21
4.2.1.2 MAX	21
4.2.2 Function Documentation	21
4.2.2.1 about()	21
4.2.2.2 amount_of_questions()	21
4.2.2.3 ask_the_audience()	22
4.2.2.4 call_a_friend()	22
4.2.2.5 check_guard()	23
4.2.2.6 create_a_single_linked_list_by_pushing_back()	23
4.2.2.7 create_list_of_players_by_pushing_front()	24
4.2.2.8 delete_list()	24
4.2.2.9 delete_list_of_players()	24
4.2.2.10 display_name()	25
4.2.2.11 fifty_fifty()	25
4.2.2.12 game_manu()	25
4.2.2.13 generate_random_number()	26
4.2.2.14 give_us_random_question()	26
4.2.2.15 how_much_money()	27
4.2.2.16 how_to_play()	27
4.2.2.17 print_list()	28
4.2.2.18 print_list_of_players()	28
4.2.2.19 read_from_file()	28
4.2.2.20 search_question()	29
4.2.2.21 sub_menu()	29
4.2.2.22 wait()	30
4.2.2.23 what_question()	30
4.2.2.24 write_to_file()	30
4.3 Milionerzy_Projekt_Marek_Kawalski/Milionerzy_Projekt_Marek_Kawalski.c File Reference	31

4.3.1 Function Documentation	31
4.3.1.1 main()	31
4.4 Milionerzy_Projekt_Marek_Kawalski/struktury.h File Reference	31
4.4.1 Macro Definition Documentation	32
4.4.1.1 _CRT_SECURE_NO_WARNINGS	32
4.4.2 Typedef Documentation	32
4.4.2.1 leaderboard	32
4.4.2.2 quest	32
4.4.3 Enumeration Type Documentation	32
4.4.3.1 bool	32
Index	35

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

leaderboard	5
question	6

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Milionerzy_Projekt_Marek_Kawalski/ funkcje.c	9
Milionerzy_Projekt_Marek_Kawalski/ funkcje.h	20
Milionerzy_Projekt_Marek_Kawalski/ Milionerzy_Projekt_Marek_Kawalski.c	31
Milionerzy_Projekt_Marek_Kawalski/ struktury.h	31

Chapter 3

Data Structure Documentation

3.1 leaderboard Struct Reference

```
#include <struktury.h>
```

Data Fields

- char `name` [50]
- char `how_much_money` [50]
- struct `leaderboard` * `pNext`

3.1.1 Detailed Description

This struct is created in order to store players names, information about their incomes during game and a pointer to the next player.

3.1.2 Field Documentation

3.1.2.1 `how_much_money`

```
char how_much_money[50]
```

3.1.2.2 `name`

```
char name[50]
```

3.1.2.3 pNext

```
struct leaderboard* pNext
```

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

- [Milionerzy_Projekt_Marek_Kawalski/struktury.h](#)

3.2 question Struct Reference

```
#include <struktury.h>
```

Data Fields

- char [question](#) [100]
- char [answer_A](#) [100]
- char [answer_B](#) [100]
- char [answer_C](#) [100]
- char [answer_D](#) [100]
- char [correct_answer](#) [10]
- int [number_of_question](#)
- bool [has_it_appeared](#)
- struct [question](#) * [pNext](#)

3.2.1 Detailed Description

This struct is used so as to have an access to data connected with questions, namely the questions themselves, a,b,c or d variants of plausible answers and finally correct answers. It also contains information if the question has already been used. I used typedef so as not to use a key word "struct" prior to each pHead pointer. It also contains a pointer to the next question.

3.2.2 Field Documentation

3.2.2.1 answer_A

```
char answer_A[100]
```

3.2.2.2 answer_B

```
char answer_B[100]
```

3.2.2.3 answer_C

```
char answer_C[100]
```

3.2.2.4 answer_D

```
char answer_D[100]
```

3.2.2.5 correct_answer

```
char correct_answer[10]
```

3.2.2.6 has_it_appeared

```
bool has_it_appeared
```

3.2.2.7 number_of_question

```
int number_of_question
```

3.2.2.8 pNext

```
struct question* pNext
```

3.2.2.9 question

```
char question[100]
```

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

- Milionerzy_Projekt_Marek_Kawalski/[struktury.h](#)

Chapter 4

File Documentation

4.1 Milionerzy_Projekt_Marek_Kawalski/funkcje.c File Reference

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

Macros

- `#define _CRT_SECURE_NO_WARNINGS`

Functions

- void `read_from_file` (quest **pHead, char *name_of_file)
- void `create_a_single_linked_list_by_pushing_back` (quest **pHead, char question[MAX], char answer_A[MAX], char answer_B[MAX], char answer_C[MAX], char answer_D[MAX], char correct_answer[10])
- void `print_list` (quest *pHead)
- void `delete_list` (quest **pHead)
- quest * `search_question` (quest *pHead, int question_number)
- int `amount_of_questions` (quest *pHead)
- void `give_us_random_question` (quest **pHead, leaderboard **Head, char **filename)
- int `generate_random_number` (int number)
- void `how_much_money` (int k)
- void `what_question` (int k)
- void `display_name` ()
- void `sub_menu` (quest **pHead, leaderboard **Head, char **filename)
- void `how_to_play` ()
- void `about` ()
- void `game_manu` (quest **pHead, leaderboard **Head, char **filename)
- void `write_to_file` (leaderboard **Head, char **filename, char name[50], char money[50], int minutes, int seconds)
- void `create_list_of_players_by_pushing_front` (leaderboard **pHead, char name[50], char money[50])

- void `wait` (int seconds)
- void `call_a_friend` (quest *pHead, char name[50], int number_of_question)
- void `fifty_fifty` (quest *pHead, int number_of_question)
- void `ask_the_audience` (quest *pHead, int number_of_question)
- void `delete_list_of_players` (leaderboard **pHead)
- void `print_list_of_players` (leaderboard *pHead)
- void `check_guard` (quest **pHead)

4.1.1 Macro Definition Documentation

4.1.1.1 _CRT_SECURE_NO_WARNINGS

```
#define _CRT_SECURE_NO_WARNINGS
```

4.1.2 Function Documentation

4.1.2.1 about()

```
void about ( )
```

Simple void function which tells the user the story about original millionaires

Parameters

<i>lack</i>	of parameters
-------------	---------------

Returns

returns nothing

4.1.2.2 amount_of_questions()

```
int amount_of_questions (
    quest * pHead )
```

Function calculates amount of questions in the list.

Parameters

<i>pHead</i>	pointer to the list
--------------	---------------------

Returns

returns amount of questions

4.1.2.3 ask_the_audience()

```
void ask_the_audience (
    quest * pHead,
    int number_of_question )
```

Function is used as a ask_the_audience lifeline.

Parameters

<i>pHead</i>	pointer to the beginning of the list
<i>number_of_question</i>	it's current's question number

Returns

returns nothing

4.1.2.4 call_a_friend()

```
void call_a_friend (
    quest * pHead,
    char name[50],
    int number_of_question )
```

Function is used as a call a friend lifeline.

Parameters

<i>pHead</i>	pointer to the beginning of the list
<i>name</i>	user's name
<i>number_of_question</i>	it's current's question number

Returns

returns nothing

4.1.2.5 check_guard()

```
void check_guard (
    quest ** pHead )
```

It's an important function which checks whether all of the questions have been used or not. If yes, it automatically turns them all into unused questions and therefore questions can repeat after the limit is exceeded.

Parameters

<i>Head</i>	pointer to the pointer to the beginning of the list (original value)
-------------	--

Returns

returns nothing

4.1.2.6 create_a_single_linked_list_by_pushing_back()

```
void create_a_single_linked_list_by_pushing_back (
    quest ** pHead,
    char question[MAX],
    char answer_A[MAX],
    char answer_B[MAX],
    char answer_C[MAX],
    char answer_D[MAX],
    char correct_answer[10] )
```

This function is used to create a single linked list by pushing back new elements. It's used in collaboration with `read_from_file` function

Parameters

<i>pHead</i>	Pointer to a pointer to the beginning of the list
<i>question</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>answerA</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>answerB</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>answerC</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>answerD</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>correct_answer</i> [10]	it's a string

See also

{[read_from_file](#)}

Returns

returns nothing

4.1.2.7 create_list_of_players_by_pushing_front()

```
void create_list_of_players_by_pushing_front (
    leaderboard ** pHead,
    char name[50],
    char money[50] )
```

Function creates list of names by pushing front. It could have been implemented in a similar way to `create_a_single_linked_list_by_pushing_back` but so as to further practice my programming skills I choose another way.

Parameters

<i>Head</i>	pointer to the pointer to the beginning of the second list (also original value)
<i>name</i>	it's the players name
<i>money</i>	it's how much has the player won

4.1.2.8 delete_list()

```
void delete_list (
    quest ** pHead )
```

This function deletes list of questions and answers so as not to have memory leaks

Parameters

<i>pHead</i>	Pointer to a pointer to the beginning of the list
--------------	---

Returns

returns nothing

4.1.2.9 delete_list_of_players()

```
void delete_list_of_players (
    leaderboard ** pHead )
```

Function deletes list of players.

Parameters

<i>Head</i>	pointer to the pointer to the beginning of the second list (original value)
-------------	---

Returns

returns nothing

4.1.2.10 display_name()

```
void display_name ( )
```

Function that using '\$' characters displays "millionairs".

Parameters

<i>no</i>	parameters
-----------	------------

Returns

returns nothing

4.1.2.11 fifty_fifty()

```
void fifty_fifty (
    quest * pHead,
    int number_of_question )
```

Function is used as a fifty_fifty lifeline.

Parameters

<i>pHead</i>	pointer to the beginning of the list
--------------	--------------------------------------

Returns

returns nothing

4.1.2.12 game_manu()

```
void game_manu (
    quest ** pHead,
    leaderboard ** Head,
    char ** filename )
```

Function which gathers various captions and couple of other functions.

Parameters

<i>pHead</i>	pointer to the pointer to the beginning of the list (working on original value)
<i>Head</i>	pointer to the pointer to the beginning of the second list (also original value)
<i>filename</i>	original file

Returns

returns nothing

4.1.2.13 generate_random_number()

```
int generate_random_number (
    int number )
```

Simple function which generates integer random number.

Parameters

<i>number</i>	it's the maximum value that can be generated
---------------	--

Returns

returns random number

4.1.2.14 give_us_random_question()

```
void give_us_random_question (
    quest ** pHead,
    leaderboard ** Head,
    char ** filename )
```

This is the foremost function in which most of other functions is executed.

Parameters

<i>pHead</i>	pointer to the pointer to the beginning of the list (working on original value)
<i>Head</i>	pointer to the pointer to the beginning of the second list (also original value)

See also

[{search_question}](#)
[{amount_of_questions}](#)

```
{generate_random_number}  
{how_much_money}  
{what_question}  
{call_a_friend}  
{fifty_fifty}  
{ask_the_audience}
```

Returns

returns nothing

4.1.2.15 how_much_money()

```
void how_much_money (  
    int k )
```

Void function which assignment is to display the amount of money the player can win.

Parameters

<i>k</i>	parameter which is essential to know what amount to money display with each question.
----------	---

Returns

returns nothing

4.1.2.16 how_to_play()

```
void how_to_play ( )
```

Simple void function which tells the user how to play.

Parameters

<i>lack</i>	of parameters
-------------	---------------

Returns

returns nothing

4.1.2.17 print_list()

```
void print_list (
    quest * pHead )
```

Additional function which was used to check if both creating and deleting the list works properly

Parameters

<i>pHead</i>	pointer to the beginning of the list
--------------	--------------------------------------

Returns

returns nothing

4.1.2.18 print_list_of_players()

```
void print_list_of_players (
    leaderboard * pHead )
```

A test function which prints list's of players content.

Parameters

<i>pHead</i>	pointer to the beginning of the list
--------------	--------------------------------------

Returns

returns nothing

4.1.2.19 read_from_file()

```
void read_from_file (
    quest ** pHead,
    char * name_of_file )
```

Function is used to read line by line data from file using gets function. All data is stored in one directional list which is created by "create_a_single_linked_list_by_pushing_back" function".

Parameters

<i>pHead</i>	Pointer to a pointer to the beginning of the list
<i>name_of_file</i>	name of the file where there are questions and answers

See also

[{create_a_single_linked_list_by_pushing_back}](#)

Returns

returns nothing

4.1.2.20 search_question()

```
quest* search_question (
    quest * pHead,
    int question_number )
```

This function searches question by the number of question and returns pointer to that question. Its used in give_↔
us_random_question function.

Parameters

<i>pHead</i>	pointer to the beginning of the list
<i>question_number</i>	int parameter

Returns

returns pointer to to the question

4.1.2.21 sub_menu()

```
void sub_menu (
    quest ** pHead,
    leaderboard ** Head,
    char ** filename )
```

Vital function which gives the player a choice of what to do and then executes adequate functions.

Parameters

<i>pHead</i>	pointer to the pointer to the beginning of the list (working on original value)
<i>Head</i>	pointer to the pointer to the beginning of the second list (also original value)
<i>filename</i>	original file

Returns

returns nothing

4.1.2.22 wait()

```
void wait (
    int seconds )
```

Function reads system time and makes the programm wait by the time specified by parameter.

Parameters

<i>seconds</i>	how long the programm is held waiting
----------------	---------------------------------------

Returns

returns nothing

4.1.2.23 what_question()

```
void what_question (
    int k )
```

Void function which is similar to "how_much_money" function. The only diffrence is that it's displayed every time disregard whether the answer had been correct or not.

Parameters

<i>k</i>	parameter which is crucial to know what amount to money display with each question.
----------	---

Returns

returns nothing

4.1.2.24 write_to_file()

```
void write_to_file (
    leaderboard ** pHead,
    char ** filename,
    char name[50],
    char money[50],
    int minutes,
    int seconds )
```

Function is used to open file in append mode.

Parameters

<i>pHead</i>	pointer to the pointer to the beginning of the second list (also original value)
--------------	--

Parameters

<i>filename</i>	original file
<i>name</i>	it's the players name
<i>money</i>	it's how much has the player won
<i>minutes</i>	it's how long has the game lasted
<i>seconds</i>	it's how long has the game lasted

Returns

returns nothing

4.2 Milionerzy_Projekt_Marek_Kawalski/funkcje.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "struktury.h"
```

Macros

- `#define _CRT_SECURE_NO_WARNINGS`
- `#define MAX 100`

Functions

- void `read_from_file` (quest **pHead, char *name_of_file)
- void `print_list` (quest *pHead)
- void `create_a_single_linked_list_by_pushing_back` (quest **pHead, char question[MAX], char answer_A[MAX], char answer_B[MAX], char answer_C[MAX], char answer_D[MAX], char correct_answer[10])
- void `delete_list` (quest **pHead)
- quest * `search_question` (quest *pHead, int question_number)
- int `amount_of_questions` (quest *pHead)
- void `give_us_random_question` (quest **pHead, leaderboard **Head, char **filename)
- int `generate_random_number` (int number)
- void `how_much_money` (int k)
- void `what_question` (int k)
- void `display_name` ()
- void `sub_menu` (quest **pHead, leaderboard **Head, char **filename)
- void `how_to_play` ()
- void `about` ()
- void `game_manu` (quest **pHead, leaderboard **Head, char **filename)
- void `write_to_file` (leaderboard **pHead, char **filename, char name[50], char money[50], int minutes, int seconds)
- void `create_list_of_players_by_pushing_front` (leaderboard **pHead, char name[50], char money[50])
- void `wait` (int seconds)
- void `call_a_friend` (quest *pHead, char name[50], int number_of_question)
- void `fifty_fifty` (quest *pHead, int number_of_question)
- void `ask_the_audience` (quest *pHead, int number_of_question)
- void `delete_list_of_players` (leaderboard **pHead)
- void `print_list_of_players` (leaderboard *pHead)
- void `check_guard` (quest **pHead)

4.2.1 Macro Definition Documentation

4.2.1.1 `_CRT_SECURE_NO_WARNINGS`

```
#define _CRT_SECURE_NO_WARNINGS
```

This macro removes secure warnings.

4.2.1.2 `MAX`

```
#define MAX 100
```

This macro defines maximum value which is used in couple of functions.

See also

[{read_from_file}](#)

[{create_a_single_linked_list_by_pushing_back}](#)

4.2.2 Function Documentation

4.2.2.1 `about()`

```
void about ( )
```

Simple void function which tells the user the story about original millionairs

Parameters

<i>lack</i>	of parameters
-------------	---------------

Returns

returns nothing

4.2.2.2 `amount_of_questions()`

```
int amount_of_questions (
    quest * pHead )
```

Function calculates amount of questions in the list.

Parameters

<i>pHead</i>	pointer to the list
--------------	---------------------

Returns

returns amount of questions

4.2.2.3 ask_the_audience()

```
void ask_the_audience (
    quest * pHead,
    int number_of_question )
```

Function is used as a ask_the_audience lifeline.

Parameters

<i>pHead</i>	pointer to the beginning of the list
<i>number_of_question</i>	it's current's question number

Returns

returns nothing

4.2.2.4 call_a_friend()

```
void call_a_friend (
    quest * pHead,
    char name[50],
    int number_of_question )
```

Function is used as a call a friend lifeline.

Parameters

<i>pHead</i>	pointer to the beginning of the list
<i>name</i>	user's name
<i>number_of_question</i>	it's current's question number

Returns

returns nothing

4.2.2.5 check_guard()

```
void check_guard (
    quest ** pHead )
```

It's an important function which checks whether all of the questions have been used or not. If yes, it automatically turns them all into unused questions and therefore questions can repeat after the limit is exceeded.

Parameters

<i>Head</i>	pointer to the pointer to the beginning of the list (original value)
-------------	--

Returns

returns nothing

4.2.2.6 create_a_single_linked_list_by_pushing_back()

```
void create_a_single_linked_list_by_pushing_back (
    quest ** pHead,
    char question[MAX],
    char answer_A[MAX],
    char answer_B[MAX],
    char answer_C[MAX],
    char answer_D[MAX],
    char correct_answer[10] )
```

This function is used to create a single linked list by pushing back new elements. It's used in collaboration with `read_from_file` function

Parameters

<i>pHead</i>	Pointer to a pointer to the beginning of the list
<i>question</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>answerA</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>answerB</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>answerC</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>answerD</i> [MAX]	it's a string which size is defined at the beginning of "funkcje.h" file
<i>correct_answer</i> [10]	it's a string

See also

{[read_from_file](#)}

Returns

returns nothing

4.2.2.7 create_list_of_players_by_pushing_front()

```
void create_list_of_players_by_pushing_front (
    leaderboard ** pHead,
    char name[50],
    char money[50] )
```

Function creates list of names by pushing front. It could have been implemented in a similar way to create_a ↔ single_linked_list_by_pushing_back but so as to further practice my programming skills I choose another way.

Parameters

<i>Head</i>	pointer to the pointer to the beginning of the second list (also original value)
<i>name</i>	it's the players name
<i>money</i>	it's how much has the player won

4.2.2.8 delete_list()

```
void delete_list (
    quest ** pHead )
```

This function deletes list of questions and answers so as not to have memory leaks

Parameters

<i>pHead</i>	Pointer to a pointer to the beginning of the list
--------------	---

Returns

returns nothing

4.2.2.9 delete_list_of_players()

```
void delete_list_of_players (
    leaderboard ** pHead )
```

Function deletes list of players.

Parameters

<i>Head</i>	pointer to the pointer to the beginning of the second list (original value)
-------------	---

Returns

returns nothing

4.2.2.10 display_name()

```
void display_name ( )
```

Function that using '\$' characters displays "millionairs".

Parameters

<i>no</i>	parameters
-----------	------------

Returns

returns nothing

4.2.2.11 fifty_fifty()

```
void fifty_fifty (
    quest * pHead,
    int number_of_question )
```

Function is used as a fifty_fifty lifeline.

Parameters

<i>pHead</i>	pointer to the beginning of the list
--------------	--------------------------------------

Returns

returns nothing

4.2.2.12 game_manu()

```
void game_manu (
    quest ** pHead,
    leaderboard ** Head,
    char ** filename )
```

Function which gathers various captions and couple of other functions.

Parameters

<i>pHead</i>	pointer to the pointer to the beginning of the list (working on original value)
<i>Head</i>	pointer to the pointer to the beginning of the second list (also original value)
<i>filename</i>	original file

Returns

returns nothing

4.2.2.13 generate_random_number()

```
int generate_random_number (
    int number )
```

Simple function which generates integer random number.

Parameters

<i>number</i>	it's the maximum value that can be generated
---------------	--

Returns

returns random number

4.2.2.14 give_us_random_question()

```
void give_us_random_question (
    quest ** pHead,
    leaderboard ** Head,
    char ** filename )
```

This is the foremost function in which most of other functions is executed.

Parameters

<i>pHead</i>	pointer to the pointer to the beginning of the list (working on original value)
<i>Head</i>	pointer to the pointer to the beginning of the second list (also original value)

See also

[{search_question}](#)

[{amount_of_questions}](#)


```
{generate_random_number}  
{how_much_money}  
{what_question}  
{call_a_friend}  
{fifty_fifty}  
{ask_the_audience}
```

Returns

returns nothing

4.2.2.15 how_much_money()

```
void how_much_money (  
    int k )
```

Void function which assignment is to display the amount of money the player can win.

Parameters

<i>k</i>	parameter which is essential to know what amount to money display with each question.
----------	---

Returns

returns nothing

4.2.2.16 how_to_play()

```
void how_to_play ( )
```

Simple void function which tells the user how to play.

Parameters

<i>lack</i>	of parameters
-------------	---------------

Returns

returns nothing

4.2.2.17 print_list()

```
void print_list (
    quest * pHead )
```

Additional function which was used to check if both creating and deleting the list works properly

Parameters

<i>pHead</i>	pointer to the beginning of the list
--------------	--------------------------------------

Returns

returns nothing

4.2.2.18 print_list_of_players()

```
void print_list_of_players (
    leaderboard * pHead )
```

A test function which prints list's of players content.

Parameters

<i>pHead</i>	pointer to the beginning of the list
--------------	--------------------------------------

Returns

returns nothing

4.2.2.19 read_from_file()

```
void read_from_file (
    quest ** pHead,
    char * name_of_file )
```

Function is used to read line by line data from file using gets function. All data is stored in one directional list which is created by "create_a_single_linked_list_by_pushing_back" function".

Parameters

<i>pHead</i>	Pointer to a pointer to the beginning of the list
<i>name_of_file</i>	name of the file where there are questions and answers

See also

[{create_a_single_linked_list_by_pushing_back}](#)

Returns

returns nothing

4.2.2.20 search_question()

```
quest* search_question (
    quest * pHead,
    int question_number )
```

This function searches question by the number of question and returns pointer to that question. Its used in give_↔
us_random_question function.

Parameters

<i>pHead</i>	pointer to the beginning of the list
<i>question_number</i>	int parameter

Returns

returns pointer to to the question

4.2.2.21 sub_menu()

```
void sub_menu (
    quest ** pHead,
    leaderboard ** Head,
    char ** filename )
```

Vital function which gives the player a choice of what to do and then executes adequate functions.

Parameters

<i>pHead</i>	pointer to the pointer to the beginning of the list (working on original value)
<i>Head</i>	pointer to the pointer to the beginning of the second list (also original value)
<i>filename</i>	original file

Returns

returns nothing

4.2.2.22 wait()

```
void wait (
    int seconds )
```

Function reads system time and makes the programm wait by the time specified by parameter.

Parameters

<i>seconds</i>	how long the programm is held waiting
----------------	---------------------------------------

Returns

returns nothing

4.2.2.23 what_question()

```
void what_question (
    int k )
```

Void function which is similar to "how_much_money" function. The only diffrence is that it's displayed every time disregard whether the answer had been correct or not.

Parameters

<i>k</i>	parameter which is crucial to know what amount to money display with each question.
----------	---

Returns

returns nothing

4.2.2.24 write_to_file()

```
void write_to_file (
    leaderboard ** pHead,
    char ** filename,
    char name[50],
    char money[50],
    int minutes,
    int seconds )
```

Function is used to open file in append mode.

Parameters

<i>pHead</i>	pointer to the pointer to the beginning of the second list (also original value)
--------------	--

Parameters

<i>filename</i>	original file
<i>name</i>	it's the players name
<i>money</i>	it's how much has the player won
<i>minutes</i>	it's how long has the game lasted
<i>seconds</i>	it's how long has the game lasted

Returns

returns nothing

4.3 Milionerzy_Projekt_Marek_Kawalski/Milionerzy_Projekt_Marek_Kawalski.c File Reference

```
#include <stdlib.h>
#include <crtdbg.h>
#include <stdio.h>
#include "funkcje.h"
#include "struktury.h"
```

Functions

- int [main](#) ()

4.3.1 Function Documentation

4.3.1.1 main()

```
int main ( )
```

4.4 Milionerzy_Projekt_Marek_Kawalski/struktury.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
```

Data Structures

- struct [question](#)
- struct [leaderboard](#)

Macros

- `#define _CRT_SECURE_NO_WARNINGS`

Typedefs

- `typedef struct question quest`
- `typedef struct leaderboard leaderboard`

Enumerations

- `enum bool { TRUE = 1, FALSE = 0 }`

4.4.1 Macro Definition Documentation

4.4.1.1 _CRT_SECURE_NO_WARNINGS

```
#define _CRT_SECURE_NO_WARNINGS
```

This macro removes secure warnings.

4.4.2 Typedef Documentation

4.4.2.1 leaderboard

```
typedef struct leaderboard leaderboard
```

This struct is created in order to store players names, information about their incomes during game and a pointer to the next player.

4.4.2.2 quest

```
typedef struct question quest
```

This struct is used so as to have an access to data connected with questions, namely the questions themselves, a,b,c or d variants of plausible answers and finally correct answers. It also contains information if the question has already been used. I used typedef so as not to use a key word "struct" prior to each pHead pointer. It also contains a pointer to the next question.

4.4.3 Enumeration Type Documentation

4.4.3.1 bool

```
enum bool
```

Enumeration which defines bool type. Bool has proved to be very useful while checking if the questions have appeared.

Enumerator

TRUE	
FALSE	

Index

`_CRT_SECURE_NO_WARNINGS`

`funkcje.c`, [10](#)
`funkcje.h`, [21](#)
`struktury.h`, [32](#)

about

`funkcje.c`, [10](#)
`funkcje.h`, [21](#)

`amount_of_questions`

`funkcje.c`, [10](#)
`funkcje.h`, [21](#)

`answer_A`

question, [6](#)

`answer_B`

question, [6](#)

`answer_C`

question, [6](#)

`answer_D`

question, [7](#)

`ask_the_audience`

`funkcje.c`, [11](#)
`funkcje.h`, [22](#)

bool

`struktury.h`, [32](#)

`call_a_friend`

`funkcje.c`, [11](#)
`funkcje.h`, [22](#)

`check_guard`

`funkcje.c`, [11](#)
`funkcje.h`, [22](#)

`correct_answer`

question, [7](#)

`create_a_single_linked_list_by_pushing_back`

`funkcje.c`, [12](#)
`funkcje.h`, [23](#)

`create_list_of_players_by_pushing_front`

`funkcje.c`, [12](#)
`funkcje.h`, [23](#)

`delete_list`

`funkcje.c`, [13](#)
`funkcje.h`, [24](#)

`delete_list_of_players`

`funkcje.c`, [13](#)
`funkcje.h`, [24](#)

`display_name`

`funkcje.c`, [14](#)
`funkcje.h`, [25](#)

FALSE

`struktury.h`, [33](#)

`fifty_fifty`

`funkcje.c`, [14](#)
`funkcje.h`, [25](#)

`funkcje.c`

`_CRT_SECURE_NO_WARNINGS`, [10](#)

about, [10](#)

`amount_of_questions`, [10](#)

`ask_the_audience`, [11](#)

`call_a_friend`, [11](#)

`check_guard`, [11](#)

`create_a_single_linked_list_by_pushing_back`, [12](#)

`create_list_of_players_by_pushing_front`, [12](#)

`delete_list`, [13](#)

`delete_list_of_players`, [13](#)

`display_name`, [14](#)

`fifty_fifty`, [14](#)

`game_manu`, [14](#)

`generate_random_number`, [15](#)

`give_us_random_question`, [15](#)

`how_much_money`, [16](#)

`how_to_play`, [16](#)

`print_list`, [16](#)

`print_list_of_players`, [17](#)

`read_from_file`, [17](#)

`search_question`, [18](#)

`sub_menu`, [18](#)

`wait`, [18](#)

`what_question`, [19](#)

`write_to_file`, [19](#)

`funkcje.h`

`_CRT_SECURE_NO_WARNINGS`, [21](#)

about, [21](#)

`amount_of_questions`, [21](#)

`ask_the_audience`, [22](#)

`call_a_friend`, [22](#)

`check_guard`, [22](#)

`create_a_single_linked_list_by_pushing_back`, [23](#)

`create_list_of_players_by_pushing_front`, [23](#)

`delete_list`, [24](#)

`delete_list_of_players`, [24](#)

`display_name`, [25](#)

`fifty_fifty`, [25](#)

`game_manu`, [25](#)

`generate_random_number`, [26](#)

`give_us_random_question`, [26](#)

`how_much_money`, [27](#)

`how_to_play`, [27](#)

- MAX, 21
- print_list, 27
- print_list_of_players, 28
- read_from_file, 28
- search_question, 29
- sub_menu, 29
- wait, 29
- what_question, 30
- write_to_file, 30
- game_manu
 - funkcje.c, 14
 - funkcje.h, 25
- generate_random_number
 - funkcje.c, 15
 - funkcje.h, 26
- give_us_random_question
 - funkcje.c, 15
 - funkcje.h, 26
- has_it_appeared
 - question, 7
- how_much_money
 - funkcje.c, 16
 - funkcje.h, 27
 - leaderboard, 5
- how_to_play
 - funkcje.c, 16
 - funkcje.h, 27
- leaderboard, 5
 - how_much_money, 5
 - name, 5
 - pNext, 5
 - struktury.h, 32
- main
 - Milionerzy_Projekt_Marek_Kawalski.c, 31
- MAX
 - funkcje.h, 21
- Milionerzy_Projekt_Marek_Kawalski.c
 - main, 31
- Milionerzy_Projekt_Marek_Kawalski/funkcje.c, 9
- Milionerzy_Projekt_Marek_Kawalski/funkcje.h, 20
- Milionerzy_Projekt_Marek_Kawalski/Milionerzy_Projekt_Marek_Kawalski.c, 31
- Milionerzy_Projekt_Marek_Kawalski/struktury.h, 31
- name
 - leaderboard, 5
- number_of_question
 - question, 7
- pNext
 - leaderboard, 5
 - question, 7
- print_list
 - funkcje.c, 16
 - funkcje.h, 27
- print_list_of_players
 - funkcje.c, 17
 - funkcje.h, 28
- quest
 - struktury.h, 32
- question, 6
 - answer_A, 6
 - answer_B, 6
 - answer_C, 6
 - answer_D, 7
 - correct_answer, 7
 - has_it_appeared, 7
 - number_of_question, 7
 - pNext, 7
 - question, 7
- read_from_file
 - funkcje.c, 17
 - funkcje.h, 28
- search_question
 - funkcje.c, 18
 - funkcje.h, 29
- struktury.h
 - _CRT_SECURE_NO_WARNINGS, 32
 - bool, 32
 - FALSE, 33
 - leaderboard, 32
 - quest, 32
 - TRUE, 33
- sub_menu
 - funkcje.c, 18
 - funkcje.h, 29
- TRUE
 - struktury.h, 33
- wait
 - funkcje.c, 18
 - funkcje.h, 29
- what_question
 - funkcje.c, 19
 - funkcje.h, 30
- write_to_file
 - funkcje.c, 30
 - funkcje.h, 30