Politechnika Śląska
Wydział Automatyki, Elektroniki i Informatyki

Programowanie Komputerów 2

Manager piłkarski

autor	Piotr Magiera
prowadzący	Grzegorz Kwiatkowski
rok akademicki	2019/2020
kierunek	Informatyka
rodzaj studiów	SSI
semestr	2
termin laboratorium	pt., 10:15 – 11:45
sekcja	6.2.
termin oddania sprawozdania	wrzesień 2020

1 Treść zadania

Napisać program symulujący zarządzaniem zespołem piłkarskim w trakcie sezonu ligowego z pozycji managera. Imiona i nazwiska zawodników oraz nazwy zespołów wczytać z pliku wejściowego. Umiejętności zawodników i trenera zmieniać w zależności od wyników. System ligowy to dwa razy każdy z każdym. Przed każdą kolejką generować kontuzje zawodników, użytkownik ma mieć możliwość dokonania transferu zawodnika lub trenera oraz dokonania zmian w składzie. Rezultat meczu symulowany ma być na podstawie umiejętności zawodników i trenera. Końcowy wynik zespołu wypisać do pliku wyjściowego.

2 Analiza zadania

Zagadnienie przedstawia problem modyfikacji i sortowania listy jednokierunkowej, komunikacji z użytkownikiem oraz wczytywania danych z pliku o ustalonym formacie zawartości.

2.1 Struktury danych

W programie wykorzystano listę jednokierunkową jako tabelę ligową z zespołami oraz listę jednokierunkową cykliczną, której elementy zawierały numery identyfikacyjne zespołów – lista ta służyła do generowania listy meczów w kolejkach. Każdy element listy zespołów zawiera wskaźnik na nazwę drużyny, liczbę punktów, numer identyfikacyjny, budżet, budżet na tygodniówki, informację o liczbie juniorów, tablicę zawodników składu podstawowego, tablicę rezerwowych i tablicę juniorów, trenera oraz wskaźnik na następny zespół w tabeli. Każdy element listy cyklicznej zawiera numer identyfikacyjny oraz wskaźnik na następny element listy (ostatni element – wskaźnik na pierwszy element listy). Wskaźnik na początek listy cyklicznej jest przesuwany na kolejne elementy w trakcie działania programu w celu wygenerowania meczów w kolejnych kolejkach.

2.2 Algorytmy

Sortowanie tabeli (listy jednokierunkowej) odbywa się z użyciem sortowania bąbelkowego. Do ustalania rozgrywek w danej kolejce wykorzystywany jest system kołowy, w którym tablica Bergera generowana jest za pomocą jednokierunkowej listy cyklicznej.

3 Specyfikacja zewnętrzna

Program uruchamiany jest z linii poleceń z wykorzystaniem następujących przełączników (kolejność przełączników jest dowolna):

-we plik wejściowy z imionami, nazwiskami oraz nazwami zespołów

-wy plik wyjściowy do wpisania ostatecznych wyników

-l liczba zespołów (parzysta, od 10 do 20)

-z nazwa drużyny gracza (początek wielką literą, reszta liter mała)

Plik wejściowy ma następującą postać: zawiera frazy kluczowe „Imiona:”, „Nazwiska:”, „Zespoly:”, po których występują dane nazwy rozdzielone znakami białymi, w czym imiona i nazwisk musi być przynajmniej po 22 (aby zapewnić odpowiednią różnorodność

zawodników), a nazw zespołów przynajmniej o jeden mniej niż podał użytkownik do konsoli. Wszystkie nazwy muszą zaczynać się wielką literą a kolejne litery muszą być małe (oczywiście bez polskich znaków). Przykładowy plik wejściowy:

Imiona:
Arnold
Bartosz
(…)

Nazwiska:
Nowak
Kowalski
(…)

Zespoly:
Ananasy
Banany
(…)

Plik wyjściowy ma format gratulacji dla użytkownika za zakończenie sezonu na danym miejscu i wypisania końcowego stanu jego drużyny.

Jeśli liczba zespołów nie jest liczbą naturalną parzystą od 10 do 20, wyświetli się komunikat

Liczba zespolow musi być pomiędzy 10 i 20 oraz byc parzysta

Jeśli nazwa zespołu nie zostanie podana, wyświetli się komunikat

Nie podano nazwy zespolu

Jeśli nazwa zespołu nie zaczyna się od wielkiej litery, wyświetli się komunikat

Pierwsza litera nazwy Twojego zespolu musi byc wielka

Jeśli kolejne litery nazwy zespołu nie są małe, wyświetli się komunikat

Wszystkie litery nazwy Twojego zespolu poza pierwsza musza byc male

Jeśli nie podano ścieżki do pliku wejściowego, wyświetli się komunikat

Nie podano nazwy pilku wejściowego

Jeśli nie podano ścieżki do pliku wyjściowego, wyświetli się komunikat

Nie podano nazwy pilku wyjściowego

Jeśli w pliku wejściowych brakuje którejś z fraz kluczowych, wyświetli się komunikat

*Brak frazy kluczowej ****fraza*** *w pliku wejściowym*

Jeśli podana ilość nazw jest niewystarczająca, wyświetli się komunikat

*Brak wystarczajacej ilosci ****nazwa*** *w pliku wejsciowy (powinny być ****liczba*** *)*

Jeśli pierwsza litera którejś z wczytanych nazw nie jest wielka, wyświetli się komunikat

*Pierwsza litera ****nazwa***** musi byc wielka*

Jeśli kolejne litery którejś z wczytanych nazw nie są małe, wyświetli się komunikat

*Wszystkie litery ****nazwa***** poza pierwsza musza byc male*

Jeśli podana jest nieprawidłowa ścieżka do pliku wejściowego lub system operacyjny uniemożliwia dostęp do tego pliku, wyświetli się komunikat

Podano nieprawidlowa sciezke do pliku wejsciowego

Jeśli nazwa zespołu użytkownika jest tożsama z którąś wczytaną, wyświetli się komunikat

Twoj zespol ma taka sama nazwe jak jeden z innych zespolow. Edytuj plik wejscowy z nazwami lub zmien nazwe zespołu

Na początku działania programu wyświetli się krótka instrukcja dla użytkownika. W trakcie również użytkownik będzie tekstowo informowany o możliwych do wykonania akcjach.

4 Specyfikacja wewnętrzna

Program został zrealizowany zgodnie z paradygmatem strukturalnym. W programie rozdzielono interfejs aplikacji (komunikację z użytkownikiem) od logiki aplikacji (edytowanie zawartości list jednokierunkowych).

4.1 Ogólna struktura programu

W funkcji głównej na początku wywoływana jest funkcja *srand(time(NULL))*. Następnie za pomocą funkcji *pobierz_dane* pobierane są dane podane przez użytkownika w konsoli. Wartość zwracana przez funkcję jest przypisywana do zmiennej *poprawnosc*. Jeśli zmienna *poprawnosc* ma wartość 0, program kończy swoje działanie. Następnie otwierany jest plik wejściowy i czytane są dane z pliku za pomocą funkcji *czytaj_dane_z_pliku*. Wartość zwracana przez funkcję jest przypisywana do zmiennej *poprawnosc*. Plik wejściowy jest zamykany. Jeśli zmienna *poprawnosc* ma wartość 0, program kończy swoje działanie. Następnie generowane są lista jednokierunkowa zespołów (tabela) i lista jednokierunkowa cykliczna do generowania meczów w kolejkach za pomocą funkcji *generuj_zespoly* i *generuj_liste_do_kolejek*. Kolejnym krokiem jest główna część programu czyli wywołanie funkcji *przeprowadz_lige*. Następnie otwierany jest plik wyjściowy, końcowy wynik i stan drużyny gracza są wypisywane do tegoż pliku za pomocą funkcji *wpisz_wyniki_do_pliku*. Zamykany jest plik wyjściowy, a zaalokowana dynamicznie pamięć jest zwalniana za pomocą funkcji *zwolnij_wszystko*.

4.2 Szczegółowy opis typów i funkcji

Szczegółowy opis typów i funkcji zawarty jest w załączniku.

5 Testowanie

Program został przetestowany na różnego rodzaju danych. Pliki wejściowe, których format jest nieprawidłowy, powodują wyświetlenie odpowiedniego komunikatu o błędzie. Niepoprawne dane podane do konsoli lub ich brak powodują wyświetlenie odpowiedniego komunikatu. Podawanie przez użytkownika niepożądanych wartości w trakcie działania programu powoduje wyświetlenie odpowiedniego komunikatu o błędzie i prośbę o ponowne podanie danej wartości. Podanie ścieżki do nieistniejącego pliku wyjściowego powoduje utworzenie pliku wyjściowego o podanej nazwie w podanej lokalizacji. Program został sprawdzony pod kątem wycieków pamięci.

6 Wnioski

Program ten jest programem prostym, ale zarazem nietrywialnym. Najbardziej wymagającą częścią zadania było napisanie kodu realizującego algorytm generowania meczów w danej kolejce za pomocą systemu kołowego i tabeli Bergera. Trudniejsze było także sortowanie listy jednokierunkowej, jednak tutaj pomogła wizualizacja problemu rysunkiem, aby nie pogubić się przy przestawianiu wskaźników. Program jest nastawiony na komunikację z użytkownikiem i jego aktywny udział w działaniu programu.

Dodatek
Szczegółowy opis typów i funkcji

Manager piłkarski

Wygenerowano przez Doxygen 1.8.17

Rozdział 1

Indeks struktur danych

1.1 Struktury danych

Tutaj znajdują się struktury danych wraz z ich krótkimi opisami:

Rozdział 2

Indeks plików

2.1 Lista plików

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

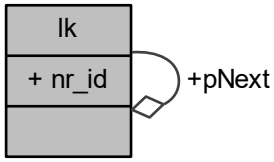
Rozdział 3

Dokumentacja struktur danych

3.1 Dokumentacja struktury lk

`#include <struktury.h>`

Diagram współpracy dla lk:



Pola danych

- int nr_id

  numer identyfikacyjny zespołu (tożsamy z numerem w strukturze zespól)
- struct lk ∗ pNext

  wskaźnik na nastepny element listy cyklicznej

3.1.1 Opis szczegółowy

Element cyklicznej listy jednokierunkowej używanej do generowania meczów w kolejkach.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- struktury.h

3.2 Dokumentacja struktury trener

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

Diagram współpracy dla trener:

Pola danych

- char * imie

  imię trenera
- char * nazwisko

  nazwisko trenera
- int umiejetnosci

  umiejętności trenera
- int klauzula_odstepnego

  kwota, za którą można kupić trenera
- int tygodniowka

  tygodniowe wynagrodzenie trenera

3.2.1 Opis szczegółowy

Struktura reprezentująca trenera klubu.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- struktury.h

3.3 Dokumentacja struktury zawodnik

`#include <struktury.h>`

Diagram współpracy dla zawodnik:

Pola danych

- char ∗ imie

  imię zawodnika

- char ∗ nazwisko

  nazwisko zawodnika

- int umiejetnosci

  umiejętności zawodnika

- int kontuzja

  wartość okreslająca, czy zawodnik jest kontuzjowany: 0 - nie, 1 - tak

- int klauzula_odstepnego

  kwota, za którą można kupić zawodnika

- int tygodniowka

  tygodniowe wynagrodzenie zawodnika

3.3.1 Opis szczegółowy

Struktura reprezentująca zawodnika grajacego w klubie.

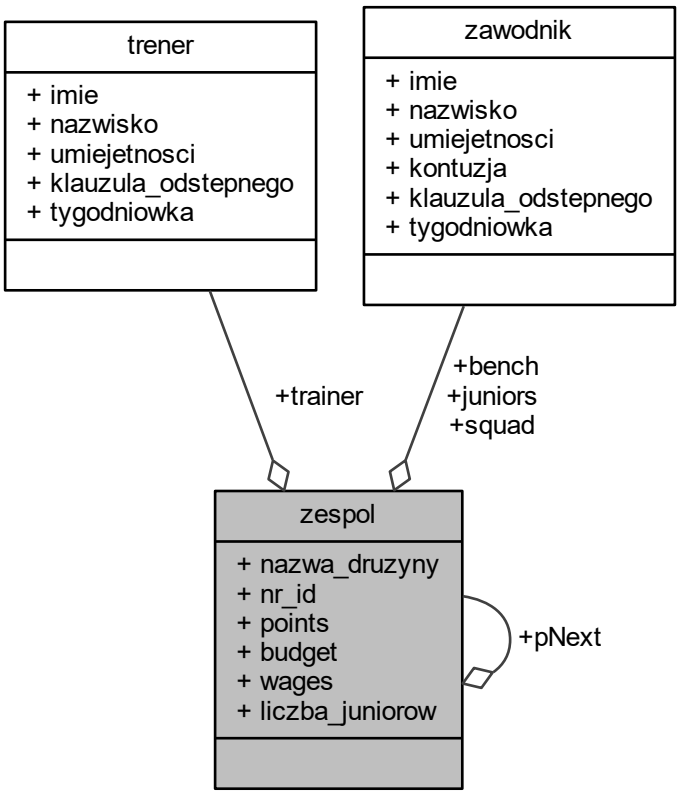
Dokumentacja dla tej struktury została wygenerowana z pliku:

- struktury.h

3.4 Dokumentacja struktury zespol

`#include <struktury.h>`

Diagram współpracy dla zespol:



Pola danych

- char * nazwa_druzyny

  nazwa drużyny
- int nr_id

  numer identyfikacyjny drużyny (tożsamy z numerem w strukturze lk)
- int points

  liczba punktów drużyny
- int budget

  budżet drużyny na transfery
- int wages

  budżet drużyny na tygodniówki
- int liczba_juniorow

  liczba juniorów znajdujących się obecnie w drużynie
- struct zawodnik squad [11]

tablica zawierająca zawodników składu podstawowego

- struct zawodnik bench [5]

  tablica zawierająca zawodników rezerwowych

- struct zawodnik juniors [5]

  tablica zawierająca juniorów

- struct trener trainer

  trener drużyny

- struct zespol ∗ pNext

  wskaźnik na następną drużynę w tabeli

3.4.1 Opis szczegółowy

Element listy jednokierunkowej reprezentującej tabelę zespołów.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- struktury.h

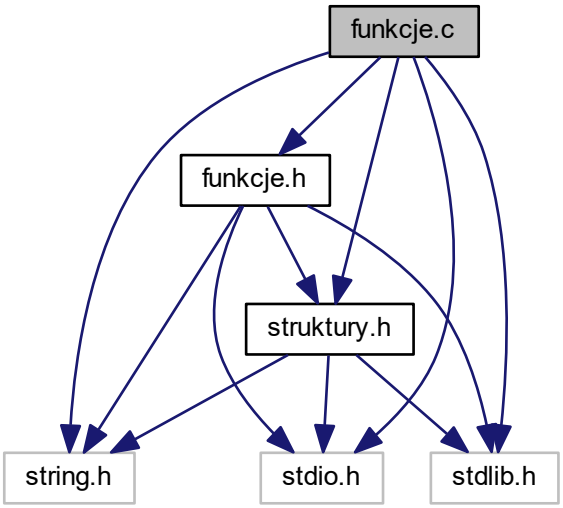
Rozdział 4

Dokumentacja plików

4.1 Dokumentacja pliku funkcje.c

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

Wykres zależności załączania dla funkcje.c:



Funkcje

- int pobierz_dane (int argc, char **argv, int *p_liczba_zespolow, char **p_nazwa_zespolu, char **p_↩ wejsciowy, char **p_wyjsciowy)

- int czytaj_nazwy (FILE **p_wejscie, char **p_nazwy, char *fraza, int liczba_nazw)
- int czytaj_dane_z_pliku (FILE **p_wejscie, int liczba_zespolow, char **imiona, char **nazwiska, char ***p_zespoly, char *nazwa_zespolu)
- void generuj_zespoly (struct zespol **ppHead, char **imiona, char **nazwiska, char **zespoly, int liczba_zespolow)
- void generuj_liste_do_kolejek (struct lk **pkpHead, struct zespol *pHead, int liczba_zespolow)
- void **generuj_juniorow** (struct zespol *pHead, char **imiona, char **nazwiska)
- void **generuj_kontuzje** (struct zespol *pHead, struct zespol *pMy)
- void sortuj_tabele (struct zespol *pHead, int liczba_zespolow)
- void symuluj_mecz (struct zespol *pT1, struct zespol *pT2)
- void umozliw_transfer (struct zespol *pHead, struct zespol *pMy, int liczba_zespolow)
- void wybierz_sklad (struct zespol *pMy)
- void przeprowadz_kolejke (struct zespol *pHead, struct lk **pkpHead, int liczba_zespolow, int nr_kolejki, char *nazwa_zespolu, char **imiona, char **nazwiska)
- void przeprowadz_lige (struct zespol *pHead, struct lk **pkpHead, int liczba_zespolow, char *nazwa_zespolu, char **imiona, char **nazwiska)
- void wypisz_zespol (FILE **p_wyjscie, struct zespol *pTemp)
- void wypisz_tabele (struct zespol *pHead)
- void wpisz_wyniki_do_pliku (FILE **p_wyjscie, struct zespol *pHead, char *nazwa_zespolu)
- void zwolnij_wszystko (struct zespol **ppHead, struct lk **pkpHead, char **imiona, char **nazwiska, char ***p_zespoly, int liczba_zespolow)

4.1.1 Dokumentacja funkcji

4.1.1.1 czytaj_dane_z_pliku()

```
int czytaj_dane_z_pliku (
            FILE ** p_wejscie,
            int liczba_zespolow,
            char ** imiona,
            char ** nazwiska,
            char *** p_zespoly,
            char * nazwa_zespolu )
```

Funkcja czyta wszystkie rodzaje danych z pliku wykorzystując funkcję czytaj_nazwy.

Parametry

p_wejscie	wskaźnik na wskaźnik do strumienia do pliku wejściowego
liczba_zespolow	liczba zespołów grających w lidze
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami
p_zespoly	wskaźnik na wskaźnik na tablicę z nazwami zespołów
nazwa_zespolu	wskaźnik na nazwę zespołu gracza

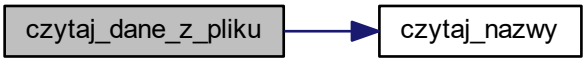
Zwraca

funkcja zwraca 0, jeżeli wczytywanie nie powiodło się prawidłowo, a 1 jeżeli wczytywanie powiodło się prawidłowo

Ostrzeżenie

funkcja alokuje pamięć

Oto graf wywołań dla tej funkcji:



4.1.1.2 czytaj_nazwy()

```
int czytaj_nazwy (
            FILE ** p_wejscie,
            char ** p_nazwy,
            char * fraza,
            int liczba_nazw )
```

Funkcja wczytuje dany rodzaj nazw z pliku wejściowego do tablicy.

Parametry

p_wejscie	wskaźnik na wskaźnik do strumienia do pliku wejściowego
p_nazwy	wskaźnik na tablicę wskaźnikow na nazwy danego typu
fraza	fraza kluczowa, po której wystąpieniu w pliku wejściowym następuje odczytywanie nazw danego typu
liczba_nazw	liczba nazw danego typu, którą trzeba wczytać

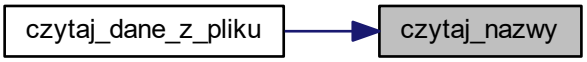
Zwraca

funkcja zwraca 0, jeżeli wczytywanie nie powiodło sie prawidłowo, a 1 jeżeli wczytywanie powiodło sie prawidłowo

Ostrzeżenie

funkcja alokuje pamięć

Oto graf wywoływań tej funkcji:



4.1.1.3 generuj_liste_do_kolejek()

```
void generuj_liste_do_kolejek (
             struct lk ** pkpHead,
             struct zespol * pHead,
             int liczba_zespolow )
```

Funkcja generuje listę cykliczną używaną do generowania meczów w kolejkach.

Parametry

pkpHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
pHead	wskaźnik na początek listy jednokierunkowej zespołów
liczba_zespolow	liczba zespołów grających w lidze

Ostrzeżenie

funkcja alokuje pamięć

4.1.1.4 generuj_zespoly()

```
void generuj_zespoly (
             struct zespol ** ppHead,
             char ** imiona,
             char ** nazwiska,
             char ** zespoly,
             int liczba_zespolow )
```

Funkcja generuje listę jednokierunkowa zespołów, służąca jako tabela.

Parametry

ppHead	wskaźnik na wskaźnik na początek listy jednokierunkowej zespołów
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami
zespoly	wskaźnik na tablicę z nazwami zespołów
liczba_zespolow	liczba zespołów grających w lidze

Ostrzeżenie

funkcja alokuje pamięć

4.1.1.5 pobierz_dane()

```
int pobierz_dane (
            int argc,
            char ** argv,
            int * p_liczba_zespolow,
            char ** p_nazwa_zespolu,
            char ** p_wejsciowy,
            char ** p_wyjsciowy )
```

Funkcja pobiera dane z konsoli i wczytuje je do zmiennych.

Parametry

argc	liczba argumentów podanych w konsoli
argv	tablica wskaźników na argumenty w konsoli
p_liczba_zespolow	wskaźnik na liczbę zespołów grających w lidze
p_nazwa_zespolu	wskaźnik na nazwę zespołu gracza
p_wejsciowy	wskaźnik na nazwę ścieżki do pliku wejściowego
p_wyjsciowy	wskaźnik na nazwę ścieżki do pliku wyjściowego

Zwraca

funkcja zwraca 0, jeżeli pobieranie danych nie powiodło sie prawidłowo, a 1 jeżeli wczytywanie danych powiodło sie prawidłowo

4.1.1.6 przeprowadz_kolejke()

```
void przeprowadz_kolejke (
            struct zespol * pHead,
            struct lk ** pkpHead,
            int liczba_zespolow,
```

```
            int nr_kolejki,
            char * nazwa_zespolu,
            char ** imiona,
            char ** nazwiska )
```

Funkcja przeprowadza kolejkę ligi.

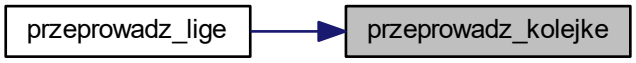
Parametry

pHead	wskaźnik na początek listy jednokierunkowej zespołów
pkpHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
liczba_zespolow	liczba zespołów grających w lidze
nr_kolejki	numer przeprowadzanej kolejki
nazwa_zespolu	wskaźnik na nazwę zespołu gracza
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami

Ostrzeżenie

funkcja może alokować pamięć

Oto graf wywoływań tej funkcji:



4.1.1.7 przeprowadz_lige()

```
void przeprowadz_lige (
            struct zespol * pHead,
            struct lk ** pkpHead,
            int liczba_zespolow,
            char * nazwa_zespolu,
            char ** imiona,
            char ** nazwiska )
```

Funkcja przeprowadza cały sezon ligi.

Parametry

pHead	wskaźnik na początek listy jednokierunkowej zespołów

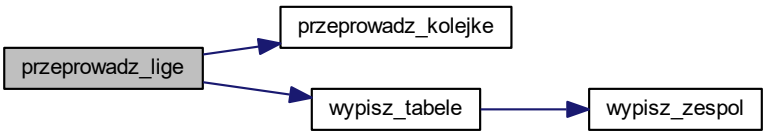
Parametry

pkpHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
liczba_zespolow	liczba zespołów grających w lidze
nazwa_zespolu	wskaźnik na nazwę zespołu gracza
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami

Ostrzeżenie

funkcja może alokować pamięć

Oto graf wywołań dla tej funkcji:



4.1.1.8 sortuj_tabele()

```
void sortuj_tabele (
            struct zespol * pHead,
            int liczba_zespolow )
```

Funkcja sortuje tabelę zespołów wedlug kolejności punktów i umiejętności zawodników z odpowiednią wagą.

Parametry

pHead	wskaznik na początek listy jednokierunkowej zespołów
liczba_zespolow	liczba zespołów grających w lidze

4.1.1.9 symuluj_mecz()

```
void symuluj_mecz (
            struct zespol * pT1,
            struct zespol * pT2 )
```

Funkcja symuluje mecz pomiędzy dwoma drużynami.

Parametry

pT1	wskaźnik na pierwszy z zespołów
pT2	wskaźnik na drugi z zespołów

4.1.1.10 umozliw_transfer()

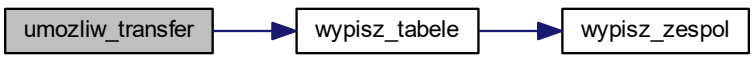
```
void umozliw_transfer (
            struct zespol * pHead,
            struct zespol * pMy,
            int liczba_zespolow )
```

Funkcja umożliwia transfer zawodników lub trenerów pomiędzy zespołami.

Parametry

pHead	wskaźnik na początek listy jednokierunkowej zespołów
pMy	wskaźnik na zespół gracza w tabeli
liczba_zespolow	liczba zespołów grających w lidze

Oto graf wywołań dla tej funkcji:



4.1.1.11 wpisz_wyniki_do_pliku()

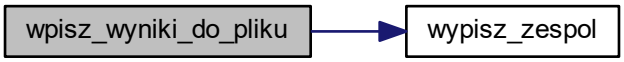
```
void wpisz_wyniki_do_pliku (
            FILE ** p_wejscie,
            struct zespol * pHead,
            char * nazwa_zespolu )
```

Funkcja wpisuje ostateczne wyniki drużyny gracza do pliku.

Parametry

p_wejscie	wskaźnik na wskaźnik do strumienia do wyjścia
pHead	wskaźnik na początek listy jednokierunkowej zespołów
nazwa_zespolu	wskaźnik na nazwę zespołu gracza

Oto graf wywołań dla tej funkcji:



4.1.1.12 wybierz_sklad()

```
void wybierz_sklad (
            struct zespol * pMy )
```

Funkcja pozwala użytkownikowi przeprowadzać zmiany w składzie.

Parametry

pMy	wskaźnik na zespół gracza w tabeli

Oto graf wywołań dla tej funkcji:

4.1.1.13 wypisz_tabele()

```
void wypisz_tabele (
            struct zespol * pHead )
```

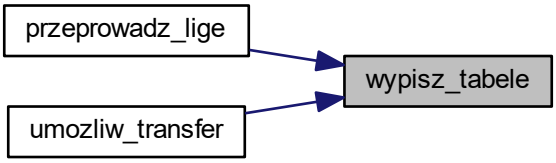
Funkcja wypisuje tabelę ligi na standardowe wyjście.

Parametry

pHead	wskaźnik na początek listy jednokierunkowej zespołów

Oto graf wywołań dla tej funkcji:

Oto graf wywoływań tej funkcji:



4.1.1.14 wypisz_zespol()

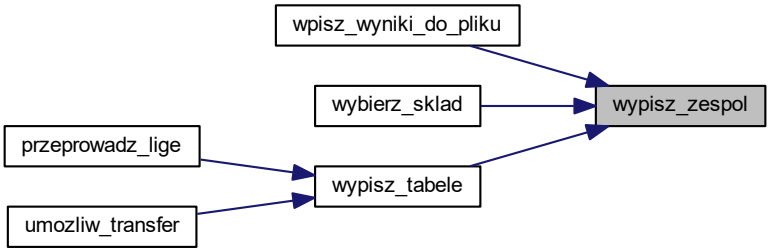
```
void wypisz_zespol (
            FILE ** p_wyjscie,
            struct zespol * pTemp )
```

Funkcja wypisuje zespół na zadane wyjście.

Parametry

p_wejscie	wskaźnik na wskaźnik do strumienia do wyjścia
pTemp	wskaźnik na wypisywany zespół w tabeli

Oto graf wywoływań tej funkcji:



4.1.1.15 zwolnij_wszystko()

```
void zwolnij_wszystko (
            struct zespol ** ppHead,
            struct lk ** pkpHead,
            char ** imiona,
            char ** nazwiska,
            char *** p_zespoly,
            int liczba_zespolow )
```

Funkcja zwalnia zaalokowaną wcześniej w programie pamięć.

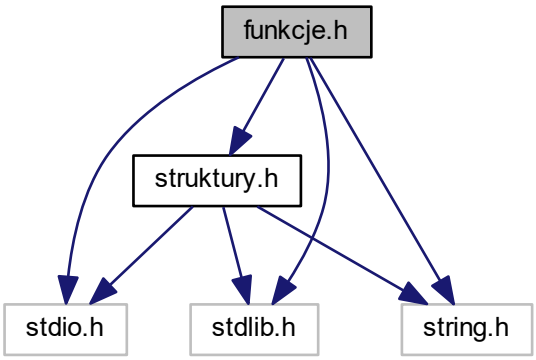
Parametry

ppHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
pkpHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami
p_zespoly	wskaźnik na wskaźnik na tablicę z nazwami zespołów
liczba_zespolow	liczba zespołów grających w lidze

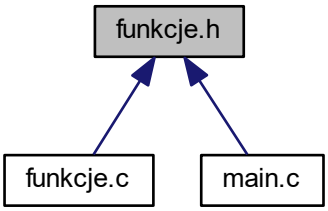
4.2 Dokumentacja pliku funkcje.h

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

Wykres zależności załączania dla funkcje.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Funkcje

- int pobierz_dane (int argc, char **argv, int *p_liczba_zespolow, char **p_nazwa_zespolu, char **p_↩ wejsciowy, char **p_wyjsciowy)
- int czytaj_nazwy (FILE **p_wejscie, char **p_nazwy, char *fraza, int liczba_nazw)
- int czytaj_dane_z_pliku (FILE **p_wejscie, int liczba_zespolow, char **imiona, char **nazwiska, char ***p_zespoly, char *nazwa_zespolu)
- void generuj_zespoly (struct zespol **ppHead, char **imiona, char **nazwiska, char **zespoly, int liczba↩ _zespolow)
- void generuj_liste_do_kolejek (struct lk **pkpHead, struct zespol *pHead, int liczba_zespolow)
- void sortuj_tabele (struct zespol *pHead, int liczba_zespolow)
- void symuluj_mecz (struct zespol *pT1, struct zespol *pT2)
- void umozliw_transfer (struct zespol *pHead, struct zespol *pMy, int liczba_zespolow)
- void wybierz_sklad (struct zespol *pMy)
- void przeprowadz_kolejke (struct zespol *pHead, struct lk **pkpHead, int liczba_zespolow, int nr_kolejki, char *nazwa_zespolu, char **imiona, char **nazwiska)

- void przeprowadz_lige (struct zespol *pHead, struct lk **pkpHead, int liczba_zespolow, char *nazwa_↩ zespolu, char **imiona, char **nazwiska)
- void wypisz_zespol (FILE **p_wyjscie, struct zespol *pTemp)
- void wypisz_tabele (struct zespol *pHead)
- void wpisz_wyniki_do_pliku (FILE **p_wejscie, struct zespol *pHead, char *nazwa_zespolu)
- void zwolnij_wszystko (struct zespol **ppHead, struct lk **pkpHead, char **imiona, char **nazwiska, char ***p_zespoly, int liczba_zespolow)

4.2.1 Dokumentacja funkcji

4.2.1.1 czytaj_dane_z_pliku()

```
int czytaj_dane_z_pliku (
            FILE ** p_wejscie,
            int liczba_zespolow,
            char ** imiona,
            char ** nazwiska,
            char *** p_zespoly,
            char * nazwa_zespolu )
```

Funkcja czyta wszystkie rodzaje danych z pliku wykorzystując funkcję czytaj_nazwy.

Parametry

p_wejscie	wskaźnik na wskaźnik do strumienia do pliku wejściowego
liczba_zespolow	liczba zespołów grających w lidze
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami
p_zespoly	wskaźnik na wskaźnik na tablicę z nazwami zespołów
nazwa_zespolu	wskaźnik na nazwę zespołu gracza

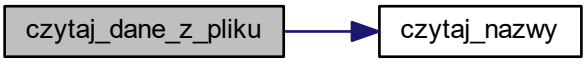
Zwraca

funkcja zwraca 0, jeżeli wczytywanie nie powiodło się prawidłowo, a 1 jeżeli wczytywanie powiodło się prawidłowo

Ostrzeżenie

funkcja alokuje pamięć

Oto graf wywołań dla tej funkcji:



4.2.1.2 czytaj_nazwy()

```
int czytaj_nazwy (
            FILE ** p_wejscie,
            char ** p_nazwy,
            char * fraza,
            int liczba_nazw )
```

Funkcja wczytuje dany rodzaj nazw z pliku wejściowego do tablicy.

Parametry

p_wejscie	wskaźnik na wskaźnik do strumienia do pliku wejściowego
p_nazwy	wskaźnik na tablicę wskaźnikow na nazwy danego typu
fraza	fraza kluczowa, po której wystąpieniu w pliku wejściowym następuje odczytywanie nazw danego typu
liczba_nazw	liczba nazw danego typu, którą trzeba wczytać

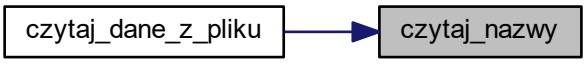
Zwraca

funkcja zwraca 0, jeżeli wczytywanie nie powiodło sie prawidłowo, a 1 jeżeli wczytywanie powiodło sie prawidłowo

Ostrzeżenie

funkcja alokuje pamięć

Oto graf wywoływań tej funkcji:



4.2.1.3 generuj_liste_do_kolejek()

```
void generuj_liste_do_kolejek (
            struct lk ** pkpHead,
            struct zespol * pHead,
            int liczba_zespolow )
```

Funkcja generuje listę cykliczną używaną do generowania meczów w kolejkach.

Parametry

pkpHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
pHead	wskaźnik na początek listy jednokierunkowej zespołów
liczba_zespolow	liczba zespołów grających w lidze

Ostrzeżenie

funkcja alokuje pamięć

4.2.1.4 generuj_zespoly()

```
void generuj_zespoly (
            struct zespol ** ppHead,
            char ** imiona,
            char ** nazwiska,
            char ** zespoly,
            int liczba_zespolow )
```

Funkcja generuje listę jednokierunkowa zespołów, służąca jako tabela.

Parametry

ppHead	wskaźnik na wskaźnik na początek listy jednokierunkowej zespołów
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami
zespoly	wskaźnik na tablicę z nazwami zespołów
liczba_zespolow	liczba zespołów grających w lidze

Ostrzeżenie

funkcja alokuje pamięć

4.2.1.5 pobierz_dane()

```
int pobierz_dane (
            int argc,
            char ** argv,
            int * p_liczba_zespolow,
            char ** p_nazwa_zespolu,
            char ** p_wejsciowy,
            char ** p_wyjsciowy )
```

Funkcja pobiera dane z konsoli i wczytuje je do zmiennych.

Parametry

argc	liczba argumentów podanych w konsoli
argv	tablica wskaźników na argumenty w konsoli
p_liczba_zespolow	wskaźnik na liczbę zespołów grających w lidze
p_nazwa_zespolu	wskaźnik na nazwę zespołu gracza
p_wejsciowy	wskaźnik na nazwę ścieżki do pliku wejściowego
p_wyjsciowy	wskaźnik na nazwę ścieżki do pliku wyjściowego

Zwraca

funkcja zwraca 0, jeżeli pobieranie danych nie powiodło sie prawidłowo, a 1 jeżeli wczytywanie danych powiodło sie prawidłowo

4.2.1.6 przeprowadz_kolejke()

```
void przeprowadz_kolejke (
            struct zespol * pHead,
            struct lk ** pkpHead,
            int liczba_zespolow,
            int nr_kolejki,
            char * nazwa_zespolu,
            char ** imiona,
            char ** nazwiska )
```

Funkcja przeprowadza kolejkę ligi.

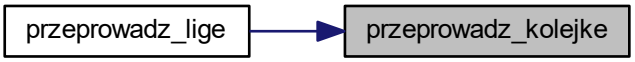
Parametry

pHead	wskaźnik na początek listy jednokierunkowej zespołów
pkpHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
liczba_zespolow	liczba zespołów grających w lidze
nr_kolejki	numer przeprowadzanej kolejki
nazwa_zespolu	wskaźnik na nazwę zespołu gracza
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami

Ostrzeżenie

funkcja może alokować pamięć

Oto graf wywoływań tej funkcji:



4.2.1.7 przeprowadz_lige()

```
void przeprowadz_lige (
            struct zespol * pHead,
            struct lk ** pkpHead,
            int liczba_zespolow,
            char * nazwa_zespolu,
            char ** imiona,
            char ** nazwiska )
```

Funkcja przeprowadza cały sezon ligi.

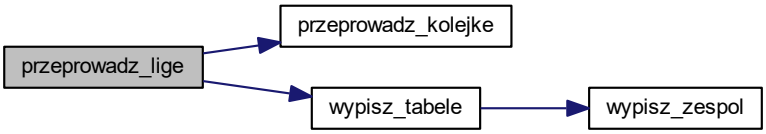
Parametry

pHead	wskaźnik na początek listy jednokierunkowej zespołów
pkpHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
liczba_zespolow	liczba zespołów grających w lidze
nazwa_zespolu	wskaźnik na nazwę zespołu gracza
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami

Ostrzeżenie

funkcja może alokować pamięć

Oto graf wywołań dla tej funkcji:



4.2.1.8 sortuj_tabele()

```
void sortuj_tabele (
            struct zespol * pHead,
            int liczba_zespolow )
```

Funkcja sortuje tabelę zespołów wedlug kolejności punktów i umiejętności zawodników z odpowiednią wagą.

Parametry

pHead	wskaznik na początek listy jednokierunkowej zespołów
liczba_zespolow	liczba zespołów grających w lidze

4.2.1.9 symuluj_mecz()

```
void symuluj_mecz (
            struct zespol * pT1,
            struct zespol * pT2 )
```

Funkcja symuluje mecz pomiędzy dwoma drużynami.

Parametry

pT1	wskaźnik na pierwszy z zespołów
pT2	wskaźnik na drugi z zespołów

4.2.1.10 umozliw_transfer()

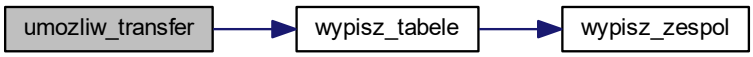
```
void umozliw_transfer (
            struct zespol * pHead,
            struct zespol * pMy,
            int liczba_zespolow )
```

Funkcja umożliwia transfer zawodników lub trenerów pomiędzy zespołami.

Parametry

pHead	wskaźnik na początek listy jednokierunkowej zespołów
pMy	wskaźnik na zespół gracza w tabeli
liczba_zespolow	liczba zespołów grających w lidze

Oto graf wywołań dla tej funkcji:



4.2.1.11 wpisz_wyniki_do_pliku()

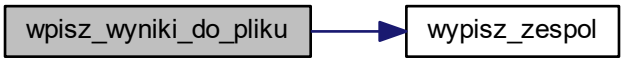
```
void wpisz_wyniki_do_pliku (
            FILE ** p_wejscie,
            struct zespol * pHead,
            char * nazwa_zespolu )
```

Funkcja wpisuje ostateczne wyniki drużyny gracza do pliku.

Parametry

p_wejscie	wskaźnik na wskaźnik do strumienia do wyjścia
pHead	wskaźnik na początek listy jednokierunkowej zespołów
nazwa_zespolu	wskaźnik na nazwę zespołu gracza

Oto graf wywołań dla tej funkcji:



4.2.1.12 wybierz_sklad()

```
void wybierz_sklad (
            struct zespol * pMy )
```

Funkcja pozwala użytkownikowi przeprowadzać zmiany w składzie.

Parametry

pMy	wskaźnik na zespół gracza w tabeli

Oto graf wywołań dla tej funkcji:

4.2.1.13 wypisz_tabele()

```
void wypisz_tabele (
            struct zespol * pHead )
```

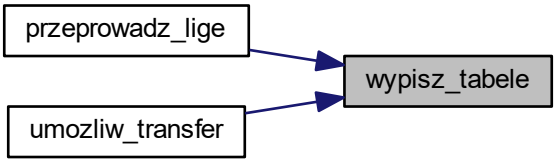
Funkcja wypisuje tabelę ligi na standardowe wyjście.

Parametry

pHead	wskaźnik na początek listy jednokierunkowej zespołów

Oto graf wywołań dla tej funkcji:

Oto graf wywoływań tej funkcji:



4.2.1.14 wypisz_zespol()

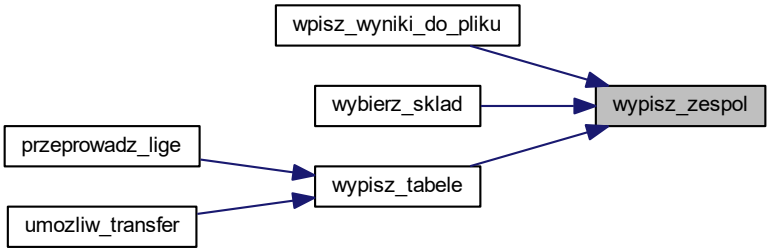
```
void wypisz_zespol (
            FILE ** p_wyjscie,
            struct zespol * pTemp )
```

Funkcja wypisuje zespół na zadane wyjście.

Parametry

p_wejscie	wskaźnik na wskaźnik do strumienia do wyjścia
pTemp	wskaźnik na wypisywany zespół w tabeli

Oto graf wywoływań tej funkcji:



4.2.1.15 zwolnij_wszystko()

```
void zwolnij_wszystko (
            struct zespol ** ppHead,
            struct lk ** pkpHead,
            char ** imiona,
            char ** nazwiska,
            char *** p_zespoly,
            int liczba_zespolow )
```

Funkcja zwalnia zaalokowaną wcześniej w programie pamięć.

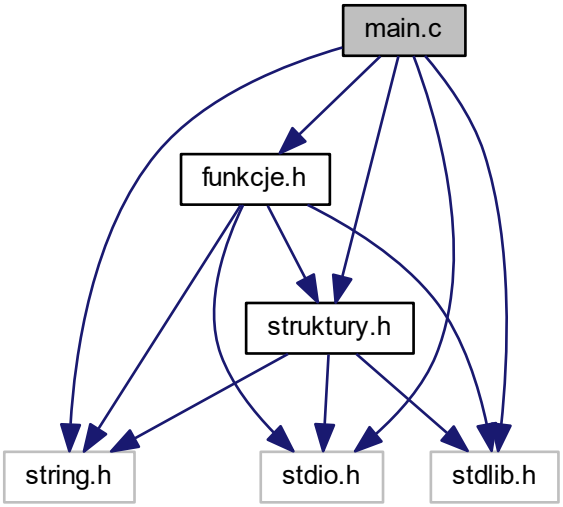
Parametry

ppHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
pkpHead	wskaźnik na wskaźnik na początek listy cyklicznej używanej do generowania meczów w kolejkach
imiona	wskaźnik na tablicę z imionami
nazwiska	wskaźnik na tablicę z nazwiskami
p_zespoly	wskaźnik na wskaźnik na tablicę z nazwami zespołów
liczba_zespolow	liczba zespołów grających w lidze

4.3 Dokumentacja pliku main.c

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

```
#include "funkcje.h"
```
Wykres zależności załączania dla main.c:

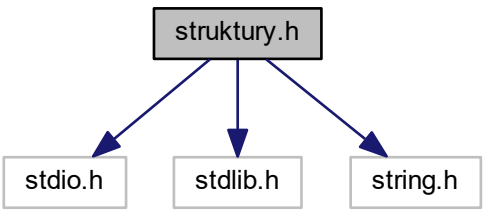


Funkcje

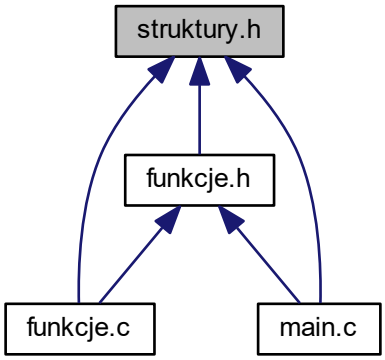
- int **main** (int argc, char ∗∗argv)

4.4 Dokumentacja pliku struktury.h

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```
Wykres zależności załączania dla struktury.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Struktury danych

- struct lk
- struct zawodnik
- struct trener
- struct zespol

Indeks