Politechnika Śląska
Wydział Informatyki, Elektroniki i Informatyki

Programowanie Komputerów

Symulator piłkarski

autor	Adam Warzecha
prowadzący	mgr in. Grzegorz Kwiatkowski
rok akademicki	2019/2020
kierunek	informatyka
rodzaj studiów	SSI
semestr	2
termin laboratorium	piątek, 12:00 – 13:30
sekcja	2.2
termin oddania sprawozdania	2020-07-24

1 **Analiza zadania**

Zagadnienie przedstawia problem posługiwania się strukturami, w szczególności listami oraz sortowaniem i symulacją.

1.1 Struktury danych

Struktury, które zostały wykorzystane w programie to:

lista jednokierunkowa w liście jednokierunkowej (cyklicznej) - kluby są pobieranie do listy cyklicznej i każdy klub posiada swoją listę jednokierunkową z zawodnikami.

Lista jednokierunkowa – dla przechowywania nazw pliku oraz danych zawodników.

1.2 Algorytmy

W programie jednym z ważniejszych algorytmów jest algorytm sortowania poprzez wstawienie. Ten algorytm jest przy wypisywaniu tabeli ligowej. Kluby są dodawane do l listy względem zdobytych punktów. Po wypisaniu tabeli zaalokowana pamięć zostaje zwolniona. Drugi z ważniejszych według mnie algorytmów jest algorytm, który odpowiada za symulację. Algorytm sumuję umiejętności piłkarzy każdej z drużyn. Następnie odejmuje mniejszy wynik od większego. I na bazie tego jak duża jest różnica, obu zespołom zostaje przydzielone przedziały ograniczające, ile pseudolosowo mogą zdobyć goli.

2 Specyfikacja zewnętrzna

Program jest uruchamiany poprzez Lokalny Debuger Windows w programie Visual Studio. Po uruchomieniu program pobiera dane z plików tekstowych, które są wykorzystywane podczas rozgrywki.

3 Specyfikacja wewnętrzna

Program został zrealizowany zgodnie z paradygmatem strukturalnym. W programie rozdzielono interfejs (komunikację z użytkownikiem) od logiki aplikacji (sortowania drużyn).

4 Ogólna struktura programu

W funkcji głównej wywołana jest funkcja pobierzRynek. Funkcja ta pobiera zawodników, których będzie można kupić w czasie gry. Następnie zostaje wywołana funkcja pobierzNazwyPlikow, która z pliku tekstowego NazwyPlikow.txt pobiera nazwy plików każdego zespołu. Po wywołaniu tej funkcji następuję wywołanie funkcji pobierzKluby, która pobiera dane wszystkich klubów z plików, których nazwy zostały pobrane przy wywołaniu wcześniejszej funkcji. Następnie wywołania zostaje funkcja zapiszSwojZespol, która za argument przyjmuje nazwę wybranego klubu zwróconego przez funkcję wybierzZespol i zapisuje klub który wybraliśmy do węzła twójZespol. Po zakończeniu wykonywania się funkcji zostaje wywołana funkcja menu, dzięki której możemy się poruszać po całym interfejsie,
Sprawdzać statystyki zespołu, symulować mecze i zakupywać zawodników. Gdy wybierzemy opcję exit funkcja zakończy swoje działanie. Na koniec zostaną wywołane następująco funkcje: usunRynek,
usunTwojZespol, usunNazwyPlikowDoListy oraz usunKluby. Zadaniem tych funkcji jest zwolnienie całej zaalokowanej wcześniej pamięci. Zakończenie wywołania tych funkcji kończy działanie programu.

4.1 Szczegółowy opis typów i funkcji

Szczegółowy opis typów i funkcji zawarty jest w załączniku.

5 Testowanie

Program został przetestowany na różnego rodzaju plikach. Plik pusty nie powoduje zgłoszenia błędu, ale nie zobaczymy żadnego rezultatu po wykonaniu programu. Pliki niepoprawne (niezawierające liczb, niezgodne ze specyfikacją) powodują błędne działanie programu. Program został sprawdzony pod kątem wycieków pamięci.

6 Wnioski

Program symulator piłkarski jest programem bardzo ciekawym. Napisałem prostą wersję tego programu ale uważam, że wraz ze wzrostem moich umiejętności będę wracał i próbował go ulepszać ponieważ mam wiele ciekawych pomysłów na działanie tego programu a możliwe, że nawet kiedyś wydania takiej aplikacji. Najtrudniejsze okazało się zapewnienie prawidłowego zwolnienia zaalokowanej pamięci oraz stworzenia wizji o tym jak strukturalnie program powinien być podzielony.

Literatura

[1] Krzysztof Simiński. Wykłady z podstaw programowania komputerów.

Symulator pilkarski

Generated by Doxygen 1.8.16

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

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

Chapter 3

Data Structure Documentation

3.1 kluby Struct Reference

`#include <Struktury.h>`

Data Fields

- char * nazwa
  napis przechowywany w liscie (nazwa klubu)
- char * trener
  napis przechowywany w liscie (nazwisko trenera)
- int budzet
  budzet zespolu
- int goleZ
  gole zdobyte
- int goleS
  gole stracone
- int bilans
  bilans bramkowy
- int pkt
  punkty
- int iloscMeczy
  ilosc rozegranych meczy
- struct kluby * next
  adres nastepnego klubu
- struct zawodnicy * zawHead
  adres pierwszego zawodnika z listy zawodnikow

3.1.1 Detailed Description

struktura listy jednokierunkowej (cyklicznej)

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

- C:/Users/Adam/Desktop/adam/Symulator/Manager/Manager/Struktury.h

3.2 nazwyPlikow Struct Reference

`#include <Struktury.h>`

Data Fields

- char * tab
  napis przechowawany w liscie (nazwa pliku)
- struct nazwyPlikow * next
  adres nastepnego elementu listy

3.2.1 Detailed Description

struktura listy jednokierunkowej

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

- C:/Users/Adam/Desktop/adam/Symulator/Manager/Manager/Struktury.h

3.3 twojZespol Struct Reference

`#include <Struktury.h>`

Data Fields

- char * nazwa
  napis przechowywany w liscie (nazwa klubu)
- char * trener
  napis przechowywany w liscie (nazwisko trenera)
- int budzet
  budzet zespolu
- int goleZ
  gole zdobyte
- int goleS
  gole stracone
- int bilans
  bilans bramkowy
- int pkt
  punkty
- int iloscMeczy
  ilosc rozegranych meczy
- struct zawodnicy * zawHead
  adres pierwszego zawodnika z listy zawodnikow

3.3.1 Detailed Description

struktura , w ktorej przechowywane sa dane o wybranym zespole

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

- C:/Users/Adam/Desktop/adam/Symulator/Manager/Manager/Struktury.h

3.4 zawodnicy Struct Reference

`#include <Struktury.h>`

Data Fields

- char * nazwisko
  napis przechowywany w liscie (nazwisko pilkarza)
- int moc
  punkty umiejetnosci pilkarza
- int wartosc
  wartosc rynkowa pilkarza
- struct zawodnicy * pNext
  adres nastepnego pilkarza

3.4.1 Detailed Description

struktura listy jednokiernukowej

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

- C:/Users/Adam/Desktop/adam/Symulator/Manager/Manager/Struktury.h

Chapter 4

File Documentation

4.1 C:/Users/Adam/Desktop/adam/Symulator/Manager/Manager/↩ Funkcje.h File Reference

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

Functions

- void zaladujNazwyPlikowDoListy (nazwyPlikow **pHead, char *buf)
- void pobierzNazwyPlikow (nazwyPlikow **pHead)
- void usunNazwyPlikowDoListy (nazwyPlikow **pHead)
- void zaladujRezerwowych (zawodnicy **rynekHead, char *i, int a, int b)
- void pobierzRynek (zawodnicy **rynekHead)
- void pobierzKluby (kluby **glowa, nazwyPlikow *pHead)
- void wypiszZawodnikow (zawodnicy *rynekHead, zawodnicy *zawHead, twojZespol *twojHead, kluby *glowa, int stanSezonu)
- void zwolnijTabele (kluby **tabela)
- void wypiszTabele (zawodnicy *rynekHead, twojZespol *twojHead, kluby *glowa, int stanSezonu)
- void aktualizuj (twojZespol *twojHead, kluby *glowa)
- void aktualizujZespol (twojZespol *twojHead, kluby *glowa)
- int sumujMoce (zawodnicy *zawHead)
- int losujGole (int min, int max)
- void symulujMecz (kluby *glowa, kluby *drugaGlowa)
- void symulujKolejke (kluby *glowa, int stanSezonu)
- void symulujSezon (kluby *glowa, kluby *end, int stanSezonu)
- void sSezonu (int stanSezonu, zawodnicy *rynekHead, twojZespol *twojHead, kluby *glowa)
- void team (zawodnicy *rynekHead, twojZespol *twojHead, kluby *glowa, int stanSezonu)
- void menu (zawodnicy *rynekHead, twojZespol *twojHead, kluby *glowa, int stanSezonu)
- void rozgrywka (zawodnicy *rynekHead, twojZespol *twojHead, kluby *glowa, int stanSezonu)
- twojZespol * zapiszSwojZespol (kluby *glowa)
- kluby * wybierzZespol (kluby *glowa)
- void kup (zawodnicy *rynekHead, twojZespol *twojHead, kluby *glowa, int stanSezonu)
- void wypiszRynek (zawodnicy *rynekHead, twojZespol *twojHead, kluby *glowa, int stanSezonu)

- void rules (zawodnicy ∗rynekHead, twojZespol ∗twojHead, kluby ∗glowa, int stanSezonu)
- void nieudanyZakup (zawodnicy ∗rynekHead, twojZespol ∗twojHead, kluby ∗glowa, int stanSezonu)
- void udanyZakup (zawodnicy ∗rynekHead, twojZespol ∗twojHead, kluby ∗glowa, int stanSezonu)
- void clear (zawodnicy ∗rynekHead)
- void usunKluby (kluby ∗∗glowa)
- void usunTwojZespol (twojZespol ∗∗twojHead)
- usunRynek (zawodnicy ∗∗rynekHead)

4.1.1 Function Documentation

4.1.1.1 aktualizuj()

```
void aktualizuj (
            twojZespol * twojHead,
            kluby * glowa )
```

Funkcja aktualizuje nasz zespol, pobiera dane o klubie, ktory wybralismy.

Parameters

twojHead	adres naszego zespolu
glowa	adres listy cyklicznej z klubami

4.1.1.2 aktualizujZespol()

```
void aktualizujZespol (
            twojZespol * twojHead,
            kluby * glowa )
```

Funkcja szuka naszego klubu w liscie cyklicznej z klubami i przekazuje adres naszego klubu funkcji aktualizuj.

Parameters

twojHead	adres naszego zespolu
glowa	adres listy cyklicznej z klubami

4.1.1.3 clear()

```
void clear (
            zawodnicy * rynekHead )
```

Funkcja ktora usuwa liste zawodnikow w klubie.

Parameters

rynekHead	adres listy jednokierunkowej z zawodnikami

4.1.1.4 kup()

```
void kup (
            zawodnicy * rynekHead,
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja, ktora umozliwia zakup zawodnika z listy transferowej do naszego zespolu.

Parameters

rynekHead	adres listy rynku transferowego
twojHead	adres naszego klubu
glowa	adres listy cyklicznej z klubami
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.5 losujGole()

```
int losujGole (
            int min,
            int max )
```

Funkcja losuje gole na podstawie mocy zespolu i zwraca ilosc goli strzelonych.

Parameters

min	minimalny przedzial
max	maksymalny przedzial

Returns

ilosc goli

4.1.1.6 menu()

```
void menu (
            zawodnicy * rynekHead,
```

```
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja, ktora przedstawia glowne menu rozgrywki, mozemy podczas wywolania tej funkcji przez do symulacji, menu naszego zespolu, rynku transferowego, zasad gry, badz wyjscia z gry.

Parameters

rynekHead	adres listy rynku transferowego
twojHead	adres naszego klubu
glowa	adres listy cyklicznej z klubami
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.7 nieudanyZakup()

```
void nieudanyZakup (
            zawodnicy * rynekHead,
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja informuje gracza o nieudanym zakupie pilkarza i pozwala wrocic do menu glownego.

Parameters

rynekHead	adres listy rynku transferowego
twojHead	adres naszego klubu
glowa	adres listy cyklicznej z klubami
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.8 pobierzKluby()

```
void pobierzKluby (
            kluby ** glowa,
            nazwyPlikow * pHead )
```

Funkcja pobiera z plikow wszystkie dane o klubach w sposob rekurencyjny. Funkcja alokuje pamiec. Funkcja zapisuje dane do listy cyklicznej.

Parameters

in,out	*glowa*	adres klubu (poczatkowo jest nullptr)
	pHead	adres, w ktorym jest zapisana nazwaPliku, z ktorego pobieramy dane

4.1.1.9 pobierzNazwyPlikow()

```
void pobierzNazwyPlikow (
            nazwyPlikow ** pHead )
```

Funkcja pobiera z pliku nazwe pliku a nastepnie przekazuje ta nazwe funkcji zaladujNazwyPlikowDoListy. Funkcja alokuje pamiec.

Parameters

in,out	*pHead*	adres pliku (poczatkowo jest nullptr)

4.1.1.10 pobierzRynek()

```
void pobierzRynek (
            zawodnicy ** rynekHead )
```

Funkcja pobiera z pliku zawodnikow a nastepnie przekazuje pobrane dane do funkcji zaladujRezerwowych. Funkcja alokuje pamiec.

Parameters

in,out	*rynekHead*	adres zawodnika z rynku transferowego (poczatkowo jest nullptr)

4.1.1.11 rozgrywka()

```
void rozgrywka (
            zawodnicy * rynekHead,
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja, w ktorej podczas wywolania mozemy symulowac mecz lub caly sezon albo wrocic do menu glownego.

Parameters

rynekHead	adres listy rynku transferowego
twojHead	adres naszego klubu
glowa	adres listy cyklicznej z klubami
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.12 rules()

```
void rules (
            zawodnicy * rynekHead,
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja wypisuje najwazniesze informacje o rozgrywce.

Parameters

rynekHead	adres listy rynku transferowego
twojHead	adres naszego klubu
glowa	adres listy cyklicznej z klubami
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.13 sSezonu()

```
void sSezonu (
            int stanSezonu,
            zawodnicy * rynekHead,
            twojZespol * twojHead,
            kluby * glowa )
```

Funkcja, ktora wypisuje stan rozgrywek po zakonczeniu sezonu i daje mozliwosc zaczecia nowego sezonu lub zamkniecia gry.

Parameters

stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu
rynekHead	adres listy rynku transferowego
twojHead	adres naszego klubu
glowa	adres listy cyklicznej z klubami

4.1.1.14 sumujMoce()

```
int sumujMoce (
            zawodnicy * zawHead )
```

Funkcja sumuje moce kazdego z zawodnikow i zwraca sume mocy wszystkich zawodnikow

Parameters

zawHead	adres poczatku listy z zawodnikami

Returns

suma mocy zawodnikow w zespole

4.1.1.15 symulujKolejke()

```
void symulujKolejke (
            kluby * glowa,
            int stanSezonu )
```

Funkcja symuluje cala kolejke, tak aby kazdy klub gral z kazdym.

Parameters

glowa	adres listy cyklicznej z klubami
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.16 symulujMecz()

```
void symulujMecz (
            kluby * glowa,
            kluby * drugaGlowa )
```

Funkcja odpowiada za symulowanie pojedynczego meczu.

Parameters

glowa	adres pierwszego zespolu
drugaGlowa	adres drugiego zespolu

4.1.1.17 symulujSezon()

```
void symulujSezon (
            kluby * glowa,
            kluby * end,
            int stanSezonu )
```

Funkcja, ktora symuluje caly sezon, po zakonczeniu symulacji zostaje wywolana funkcja sSezonu.

Parameters

glowa	adres listy cyklicznej z klubami
end	adres listy cyklicznej z klubami, do ktorego beda odbywaly sie symulacje
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.18 team()

```
void team (
            zawodnicy * rynekHead,
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja, ktora jest menu naszego zespolu, podczas wywolania tej funckji mozemy wywolac inne funkcje, dzieki ktorym, zobaczymy aktualna tabele, naszych zawodnikow czy wrocimy do menu glownego.

Parameters

rynekHead	adres listy rynku transferowego
twojHead	adres naszego klubu
glowa	adres listy cyklicznej z klubami
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.19 udanyZakup()

```
void udanyZakup (
            zawodnicy * rynekHead,
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja informuje gracza o udanym zakupie pilkarza i pozwala wrocic do menu glownego.

Parameters

rynekHead	adres listy rynku transferowego
twojHead	adres naszego klubu
glowa	adres listy cyklicznej z klubami
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.20 usunKluby()

```
void usunKluby (
            kluby ** glowa )
```

Funkcja, ktora usuwa wszystkie kluby z listy jednokierunkowej cyklicznej. Przed usunieciem danego klubu wywoly-wana jest funkcja usunListeZawodnikow, ktora dla kazdego klubu usuwa jego zawodnikow.

Parameters

glowa	adres listy cyklicznej jednokierunkowej z klubami

4.1.1.21 usunNazwyPlikowDoListy()

```
void usunNazwyPlikowDoListy (
            nazwyPlikow ** pHead )
```

Funkcja usuwa wszystkie nazwy plikow z listy (od poczatku, iteracyjnie).

Parameters

in,out	*pHead*	adres pierwszej nazwy pliku w liscie

4.1.1.22 usunRynek()

```
usunRynek (
            zawodnicy ** rynekHead )
```

Funkcja, usuwa zaalokowana pamiec dla zawodnikow z listy rezerwowych.

Parameters

rynekHead	poczatek listy jednokierunkowej z zawodnikami

4.1.1.23 usunTwojZespol()

```
void usunTwojZespol (
            twojZespol ** twojHead )
```

Funkcja, ktora usuwa zaalokowana pamiec w strukturze twojZespol.

Parameters

twojHead	adres naszej struktury

4.1.1.24 wybierzZespol()

```
kluby* wybierzZespol (
            kluby * glowa )
```

Funkcja, w wywolaniu ktorej, wybieramy zespol.

Parameters

glowa	adres listy cyklicznej z klubami

4.1.1.25 wypiszRynek()

```
void wypiszRynek (
            zawodnicy * rynekHead,
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja odpowiada za wypisanie listy z zawodnikami do kupienia.

Parameters

rynekHead	adres listy rynku transferowego
twojHead	adres naszego klubu
glowa	adres listy cyklicznej z klubami
stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.26 wypiszTabele()

```
void wypiszTabele (
            zawodnicy * rynekHead,
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja wypisuje obecna tabele ligowa i poszczegolne statystyki klubow. Funkcja alokuje pamiec. Sortuje tabele poprzez wstawianie. Po wypisaniu tabeli, nastepuje zwolnienie zaalokowanej pamieci. Po wypisaniu zwolnieniu zaalokowanej pamieci i wprowadzeniu w konsoli '1' wywolamy funkcje team, dzieki ktorej wrocimy do menu z naszym zespolem.

Parameters

	rynekHead	adres listy rynku transferowego
	twojHead	adres naszego klubu
in,out	*glowa*	adres listy cyklicznej z klubami
	stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.27 wypiszZawodnikow()

```
void wypiszZawodnikow (
            zawodnicy * rynekHead,
            zawodnicy * zawHead,
            twojZespol * twojHead,
            kluby * glowa,
            int stanSezonu )
```

Funkcja wypisuje zawodnikow i ich dane z naszego zespolu. Po wypisaniu zawodnikow i wprowadzeniu w konsoli '1' wywolamy funkcje team, dzieki ktorej wrocimy do menu z naszym zespolem.

Parameters

	rynekHead	adres listy rynku transferowego
in,out	*zawHead*	adres poczatku listy z zawodnikami, z ktorej bedziemy wypisywac zawodnikow
	twojHead	adres naszego klubu
	glowa	adres listy cyklicznej z klubami
	stanSezonu	parametr, ktory inforuje program ile meczy pozostalo do konca sezonu

4.1.1.28 zaladujNazwyPlikowDoListy()

```
void zaladujNazwyPlikowDoListy (
            nazwyPlikow ** pHead,
            char * buf )
```

Funkcja dodaje na poczatek listy nazwe pliku (iteracyjnie). Funkcja alokuje pamiec.

Parameters

in,out	*pHead*	adres nazwy pliku (poczatkowo jest nullptr)
	buf	nazwa pliku, ktora ma byc wstawiona do listy

4.1.1.29 zaladujRezerwowych()

```
void zaladujRezerwowych (
            zawodnicy ** rynekHead,
            char * i,
            int a,
            int b )
```

Funkcja dodaje na poczatek listy zawodnika (iteracyjnie). Funkcja alokuje pamiec.

Parameters

in,out	*rynekHead*	adres zawodnika z rynku transferowego (poczatkowo jest nullptr)
	i	nazwisko zawodnika, ktory ma byc wstawiona do listy
	a	moc zawodnika, ktora ma byc wstawiona do listy
	b	wartosc zawodnika, ktora ma byc wstawiona do listy

4.1.1.30 zapiszSwojZespol()

```
twojZespol* zapiszSwojZespol (
            kluby * glowa )
```

Funkcja, pobiera dane klubu, ktory wybralismy do struktury naszego zespolu. Funkcja alokuje pamiec.

Parameters

glowa	adres wybranego klubu w strukturze z klubami

Returns

adres naszego zespolu w strukturze z naszym klubem

4.1.1.31 zwolnijTabele()

```
void zwolnijTabele (
            kluby ** tabela )
```

Funkcja usuwa wszystkie kluby z listy (od poczatku, iteracyjnie).

Parameters

in,out	*tabela*	adres pierwszej nazwy klubu w liscie

4.2 C:/Users/Adam/Desktop/adam/Symulator/Manager/Manager/Struktury.h File Reference

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

Data Structures

- struct nazwyPlikow
- struct zawodnicy
- struct kluby
- struct twojZespol

Typedefs

- typedef struct nazwyPlikow nazwyPlikow
- typedef struct zawodnicy zawodnicy
- typedef struct kluby kluby
- typedef struct twojZespol twojZespol

4.2.1 Detailed Description

4.2.2 Typedef Documentation

4.2.2.1 kluby

```
typedef struct kluby kluby
```

struktura listy jednokierunkowej (cyklicznej)

4.2.2.2 nazwyPlikow

```
typedef struct nazwyPlikow nazwyPlikow
```

struktura listy jednokierunkowej

4.2.2.3 twojZespol

```
typedef struct twojZespol twojZespol
```

struktura , w ktorej przechowywane sa dane o wybranym zespole

4.2.2.4 zawodnicy

```
typedef struct zawodnicy zawodnicy
```

struktura listy jednokiernukowej