NAGY HÁZI FELADAT

Programozásalai
2.
Feladatspecifió

Bakó Péter

OTQVE5

TARTALOM

1. **Feladat……..2**
2. **A program célja……2**
3. **A program használata……2-3**
4. **Nyelvi elemek……...3**
5. **Megvalósítási tervezet……..3**

1. Feladat

Sportegyesület

A KSI Sportegyesület nyilvántartást szeretne vezetni a csapatairól. Minden csapat rendelkezik egy névvel és egy alaplétszámmal. A sportegyesület háromféle sportággal foglalkozik: Kajak, Kenu és labdarúgás. A kenu csapatnak két edzője van; a labdarugó csapatnak elengedhetetlen kellékei a labdák aminek létszámát is nyilvántartják; a kajak csapatok pedig évente kapnak valamekkora összegű támogatást.

2. A program célja

Egy sportegyesületnek több szakosztálya van amiknek mind különböznek a létszámaik illetve igényeik, így egy személynek nehéz ezeket a dolgokat nyilván tartania. A program célja, hogy egy olyan felületet teremtsen ahol betekintést nyerhet a felhasználó a KSI SE szakosztályaiba illetve egy személyben vezethessen nyilvántartást az aktuális csapatokról illetve azok bizonyos tulajdonságairól.

3. A program használata

Futtatva a programot a felhasználónak lehetősége lesz új Sport egyesületet létrehozni, annak nevet adni és azon belül 3 sportágak speciális adatait:

- Kenu: ennek az alosztálynak a különlegessége, hogy az itt edző gyerekeknek két edzője van, így a csapat létszáma mellett ezt is nyilván lehet tartani.
- Labdarúgás: ennek a sportágnak az elengedhetetlen követelmény, hogy számon tudjuk követni mennyi labdája van az

egyesületnek, ezt meg is tudjuk tenni a programmal, sőt a csapatunk létszámát is meg tudjuk adniilletve lekérdezni.

- Kajak: mivel Magyarország legsikeresebb sportága, így ők kapnak különböző forrásokból támogatást éves szinten, amit nyomon tudunk követni, illetve itt is megnézhetjük csapatunk létszámát és változtathatjukis azt.

A felhasználónak lehetősége lesz a összes csapat törlésére illetve azoknak listázására, továbbá kiírathatja egy file-ba (file nevét is a felhasználó adja így arra kérjük tartsa szem előtt a file név adási szabályokat) az eddig felvett csapatokat így a program későbbi megnyitásakor is be tudja őket olvasni, és így folyamatosan számon tarthatja a felhasználó akár a régebbi csapatait is ami hasznos lehet egy nagyobb csapatátalakítás alkalmával. Lehetősége van még a felhasználónak egy bizonyos csapatot is törölni a tarolóból név alapján. A program menüpontjait illetve azokon belüli navigációt a mondat/szó előtti szögletes zárójelek közötti szám lenyomásával tudjuk elérni

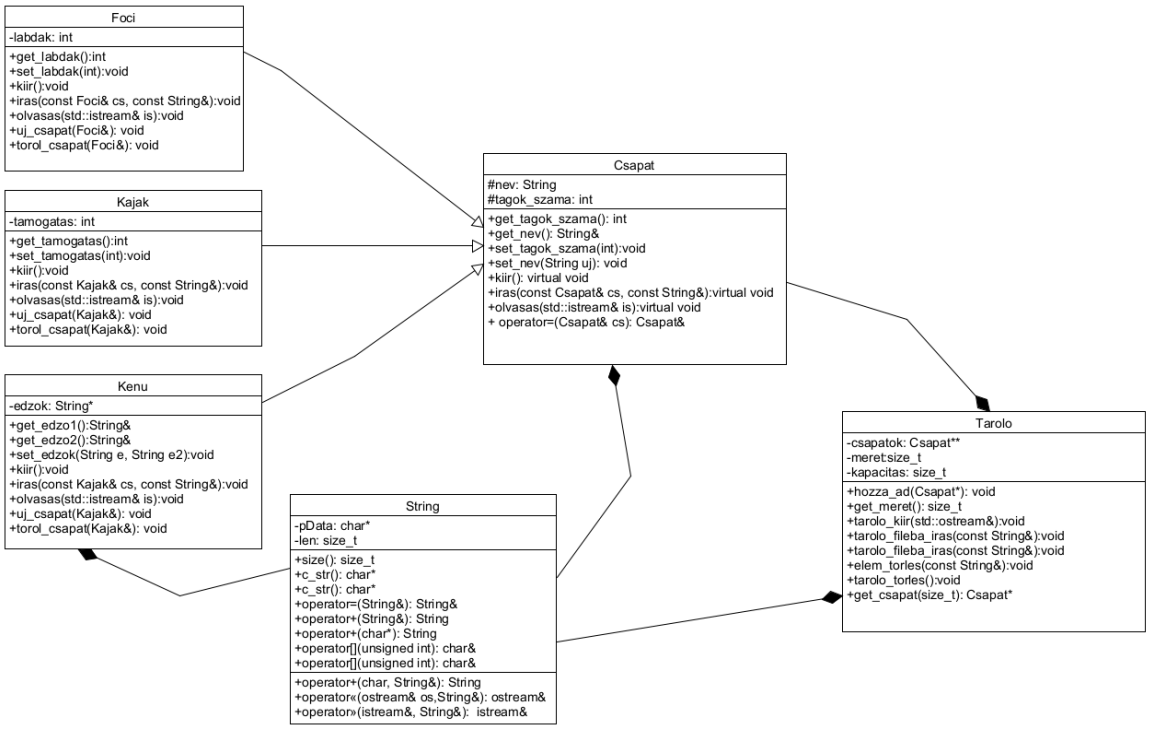
Helytelen karakter megadása, illetve helytelen formátum esetén a program figyelmezteti a felhasználót.

4. Nyelvi elemek

Az egész program kezelőfelülete magyar lesz viszont ékezetes betűk kezelésére alkalmatlan, így a felhasználónak ékezet nélküli neveket kell választania.

5. Megvalósítási tervezet

Az osztályok egymáshoz való viszonya UML-diagrammal (tervezet)



Sportegyesület

Készítette Doxygen 1.9.7

1. Hierarchikus mutató
1.1. Osztályhierarchia
2. Osztálymutató
2.1. Osztálylista
3. Fájlmutató
3.1. Fájllista
4. Osztályok dokumentációja
4.1. Csapat osztályreferencia
4.1.1. Konstruktorok és destruktorok dokumentációja
4.1.1.1. Csapat() [1/2]
4.1.1.2. Csapat() [2/2]
4.1.2. Tagfüggvények dokumentációja
4.1.2.1. iras()
4.1.2.2. kiir()
4.1.2.3. olvasas()
4.2. Foci osztályreferencia
4.2.1. Tagfüggvények dokumentációja
4.2.1.1. iras()
4.2.1.2. kiir()
4.2.1.3. olvasas()
4.3. Kajak osztályreferencia
4.3.1. Tagfüggvények dokumentációja
4.3.1.1. iras()
4.3.1.2. kiir()
4.3.1.3. olvasas()
4.4. Kenu osztályreferencia
4.4.1. Tagfüggvények dokumentációja
4.4.1.1. iras()
4.4.1.2. kiir()
4.4.1.3. olvasas()
4.5. Tarolo osztályreferencia
4.5.1. Konstruktorok és destruktorok dokumentációja
4.5.1.1. Tarolo()
4.5.2. Tagfüggvények dokumentációja
4.5.2.1. elem_torles()
4.5.2.2. get_csapat()
4.5.2.3. hozza_ad()
4.5.2.4. olvasas()
4.5.2.5. tarolo_fileba_iras()
4.5.2.6. tarolo_kiir()

5. Fájlok dokumentációja

5.1. egyesulet.h .....

5.2. egyesulet_test.h .....

5.3. tarolo.h .....

5.4. vezerles.h .....

Tárgymutató

1. fejezet

Hierarchikus mutató

1.1. Osztályhierarchia

Majdnem (de nem teljesen) betűrendbe szedett leszármazási lista:

Csapat 7
Foci 9
Kajak 11
Kenu 13
Tarolo 15

2. fejezet

Osztálymutató

2.1. Osztálylista

Az összes osztály, struktúra, unió és interfész listája rövid leírásokkal:

Csapat 7
Foci 9
Kajak 11
Kenu 13
Tarolo 15

3. fejezet

Fájlmutató

3.1. Fájllista

Az összes dokumentált fájl listája rövid leírásokkal:

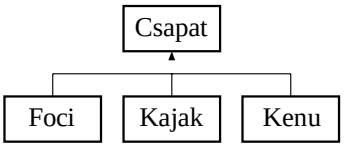
egyesulet.h 19
egyesulet_test.h 20
tarolo.h 20
vezerles.h 21

4. fejezet

Osztályok dokumentációja

4.1. Csapat osztályreferencia

A Csapat osztály származási diagramja:



Publikus tagfüggvények

- Csapat (const String nev="", size_t tagok_szama=0)
  Konstruktor.
- Csapat (Csapat &cs)
- const String & **get_nev** () const
  Getterek.
- size_t **get_tagok_szama** () const
- void **set_tagok_szama** (size_t uj)
  Setterek.
- void **set_nev** (String uj)
- virtual void kiir (std::ostream &os) const =0
  Virtualis kiiras.
- virtual void iras (std::ofstream &fout)=0
  Fajlba valo virtualis iras.
- virtual void olvasas (std::ifstream &fin)=0
  Fajlba valo virtualis iras.
- Csapat & **operator=** (Csapat &cs)
  operator =
- virtual ~**Csapat** ()
  Destruktor.

Védett attribútumok

- String **nev**
- size_t **tagok_szama**

4.1.1. Konstruktorok és destruktorok dokumentációja

4.1.1.1. Csapat() [1/2]

```
Csapat::Csapat (
            const String nev = ¨¨,
            size_t tagok_szama = 0 )  [inline]
```

Konstruktor.

Csapat osztály konstruktora

Paraméterek

nev	- a csapat neve
tagok_szama	- a csapat tagjainak száma

4.1.1.2. Csapat() [2/2]

```
Csapat::Csapat (
            Csapat & cs )  [inline]
```

Csapat osztály másoló konstruktora

Paraméterek

cs	- másolandó Csapat objektum

4.1.2. Tagfüggvények dokumentációja

4.1.2.1. iras()

```
virtual void Csapat::iras (
            std::ofstream & fout )  [pure virtual]
```

Fajlba valo virtualis iras.

Megvalósítják a következők: Foci, Kajak és Kenu.

4.1.2.2. kiir()

```
virtual void Csapat::kiir (
            std::ostream & os ) const  [pure virtual]
```

Virtualis kiiras.

Megvalósítják a következők: Foci, Kajak és Kenu.

4.1.2.3. olvasas()

```
virtual void Csapat::olvasas (
            std::ifstream & fin )  [pure virtual]
```

Fajlba valo virtualis iras.

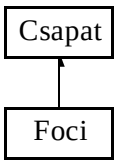
Megvalósítják a következők: Foci, Kajak és Kenu.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- egyesulet.h
- egyesulet.cpp

4.2. Foci osztályreferencia

A Foci osztály származási diagramja:



Publikus tagfüggvények

- **Foci** (const String nev="", size_t tagok=0, size_t labdak=0)
  Konstruktor.
- size_t **get_labdak** () const
  Getterek.
- void **set_labdak** (size_t uj)
  Setterek.
- void kiir (std::ostream &os) const
- void iras (std::ofstream &fout)
- void olvasas (std::ifstream &fin)
- ~**Foci** ()
  Destruktor.

Publikus tagfüggvények a(z) Csapat osztályból származnak

- Csapat (const String nev="", size_t tagok_szama=0)
  Konstruktor.
- Csapat (Csapat &cs)
- const String & **get_nev** () const
  Getterek.
- size_t **get_tagok_szama** () const
- void **set_tagok_szama** (size_t uj)
  Setterek.
- void **set_nev** (String uj)

- virtual void kiir (std::ostream &os) const =0

  Virtualis kiiras.
- virtual void iras (std::ofstream &fout)=0

  Fajlba valo virtualis iras.
- virtual void olvasas (std::ifstream &fin)=0

  Fajlba valo virtualis iras.
- Csapat & **operator=** (Csapat &cs)

  operator =
- virtual ~**Csapat** ()

  Destruktor.

További örökölt tagok

Védett attribútumok a(z) Csapat osztályból származnak

- String **nev**
- size_t **tagok_szama**

4.2.1. Tagfüggvények dokumentációja

4.2.1.1. iras()

```
void Foci::iras (
            std::ofstream & fout )  [virtual]
```

Foci osztály filba mentese

Paraméterek

fout	- ofstream típusú fajlstream, ahova kiirjuk az osztaly adatait es egy szamot ami alapjan a taroloban beolvasaskor tudni fogjuk milyen osztalyrol van szo

Megvalósítja a következőket: Csapat.

4.2.1.2. kiir()

```
void Foci::kiir (
            std::ostream & os ) const  [virtual]
```

Kiir Foci osztály kiir metódusa

Paraméterek

os	- output stream, ide irjuk ki a Kajak csapat adatait

Megvalósítja a következőket: Csapat.

4.2.1.3. olvasas()

```
void Foci::olvasas (
            std::ifstream & fin )  [virtual]
```

Foci osztály olvasas metódusa

Paraméterek

fin	- ifstream tipusú fajlstream, ahova be olvassuk a filban levo adatokat

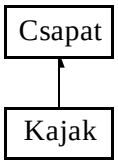
Megvalósítja a következőket: Csapat.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- egyesulet.h
- egyesulet.cpp

4.3. Kajak osztályreferencia

A Kajak osztály származási diagramja:



Publikus tagfüggvények

- **Kajak** (const String nev="", size_t tagok=0, size_t tamogatas=0)
  Konstruktor.
- size_t **get_tamogatas** () const
  Getterek.
- void **set_tamogatas** (size_t uj)
  Setterek.
- void kiir (std::ostream &os) const
- void iras (std::ofstream &fout)
- void olvasas (std::ifstream &fin)
- ~**Kajak** ()
  Destruktor.

Publikus tagfüggvények a(z) Csapat osztályból származnak

- Csapat (const String nev="", size_t tagok_szama=0)
  Konstruktor.
- Csapat (Csapat &cs)
- const String & **get_nev** () const
  Getterek.
- size_t **get_tagok_szama** () const
- void **set_tagok_szama** (size_t uj)
  Setterek.
- void **set_nev** (String uj)
- virtual void kiir (std::ostream &os) const =0
  Virtualis kiiras.
- virtual void iras (std::ofstream &fout)=0
  Fajlba valo virtualis iras.
- virtual void olvasas (std::ifstream &fin)=0
  Fajlba valo virtualis iras.
- Csapat & **operator=** (Csapat &cs)
  operator =
- virtual ~**Csapat** ()
  Destruktor.

További örökölt tagok

Védett attribútumok a(z) Csapat osztályból származnak

- String **nev**
- size_t **tagok_szama**

4.3.1. Tagfüggvények dokumentációja

4.3.1.1. iras()

```
void Kajak::iras (
            std::ofstream & fout )  [virtual]
```

Kajak osztály filba mentese

Paraméterek

fout	- ofstream típusú fajlstream, ahova kiirjuk az osztaly adatait es egy szamot ami alapjan a taroloban beolvasaskor tudni fogjuk milyen osztalyrol van szo

Megvalósítja a következőket: Csapat.

4.3.1.2. kiir()

```
void Kajak::kiir (
            std::ostream & os ) const  [virtual]
```

Kiir Kajak osztály kiir metódusa

Paraméterek

os	- output stream, ide irjuk ki a Kajak csapat adatait

Megvalósítja a következőket: Csapat.

4.3.1.3. olvasas()

```
void Kajak::olvasas (
            std::ifstream & fin )  [virtual]
```

Kajak osztály filebol beolvasasa

Paraméterek

fin	- ifstream típusú fajlstream, ahova be olvassuk a filban levo adatokat

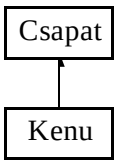
Megvalósítja a következőket: Csapat.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- egyesulet.h
- egyesulet.cpp

4.4. Kenu osztályreferencia

A Kenu osztály származási diagramja:



Publikus tagfüggvények

- **Kenu** (const String nev="", size_t tagok=0, String edzo1="", String edzo2="")
  Konstruktor.
- const String & **get_edzo1** () const
  Getterek.
- const String & **get_edzo2** () const
- void **set_edzok** (String edzo1="", String edzo2="")
  Setterek.
- void kiir (std::ostream &os) const
- void iras (std::ofstream &fout)
- void olvasas (std::ifstream &fin)
- ~**Kenu** ()
  Destruktor.

Publikus tagfüggvények a(z) Csapat osztályból származnak

- Csapat (const String nev="", size_t tagok_szama=0)
  Konstruktor.
- Csapat (Csapat &cs)
- const String & **get_nev** () const
  Getterek.
- size_t **get_tagok_szama** () const
- void **set_tagok_szama** (size_t uj)
  Setterek.
- void **set_nev** (String uj)
- virtual void kiir (std::ostream &os) const =0
  Virtualis kiiras.
- virtual void iras (std::ofstream &fout)=0
  Fajlba valo virtualis iras.
- virtual void olvasas (std::ifstream &fin)=0
  Fajlba valo virtualis iras.
- Csapat & **operator=** (Csapat &cs)
  operator =
- virtual ~**Csapat** ()
  Destruktor.

További örökölt tagok

Védett attribútumok a(z) Csapat osztályból származnak

- String **nev**
- size_t **tagok_szama**

4.4.1. Tagfüggvények dokumentációja

4.4.1.1. iras()

```
void Kenu::iras (
            std::ofstream & fout )  [virtual]
```

Kenu osztály filba mentese

Paraméterek

fout	- ofstream típusú fajlstream, ahova kiirjuk az osztaly adatait es egy szamot ami alapjan a taroloban beolvasaskor tudni fogjuk milyen osztalyrol van szo

Megvalósítja a következőket: Csapat.

4.4.1.2. kiir()

```
void Kenu::kiir (
            std::ostream & os ) const  [virtual]
```

Kiir Kenu osztály kiir metódusa

Paraméterek

os	- output stream, ide irjuk ki a Kenu csapat adatait

Megvalósítja a következőket: Csapat.

4.4.1.3. olvasas()

```
void Kenu::olvasas (
            std::ifstream & fin )  [virtual]
```

Kenu osztály filebol beolvasasa

Paraméterek

fin	- ifstream típusú fajlstream, ahova be olvassuk a filban levo adatokat

Megvalósítja a következőket: Csapat.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- egyesulet.h
- egyesulet.cpp

4.5. Tarolo osztályreferencia

Publikus tagfüggvények

- Tarolo (size_t kezdeti_kapacitas=10)
- void hozza_ad (Csapat *cs)
- size_t **get_meret** () const

  Getter.
- void tarolo_kiir (std::ostream &os) const
- void tarolo_fileba_iras (const String &filenev)
- void olvasas (const String &filenev)
- void elem_torles (const String &nev)
- void **tarolo_torles** ()

  A tarolo teljes tartalmat torli, es felszabditja.
- Csapat * get_csapat (size_t idx) const
- ~**Tarolo** ()

  Destruktor, ami felszabaditja az osszes dinamikusan foglalat memoriat.

4.5.1. Konstruktorok és destruktorok dokumentációja

4.5.1.1. Tarolo()

```
Tarolo::Tarolo (
            size_t kezdeti_kapacitas = 10 )  [inline]
```

Tarolo osztály konstruktora

Paraméterek

kezdeti_kapacitas	- kezdeti tarolo kapacitasa (alapertelmezetten 10) a meretet alapertelmezetten 0-ra allitja illetve foglal a kapacitasnak megfelelo meretu helyet a tarolonak

4.5.2. Tagfüggvények dokumentációja

4.5.2.1. elem_torles()

```
void Tarolo::elem_torles (
            const String & nev )
```

Csapat torlese a tarolobol nev alapjan

Paraméterek

nev	- ez alapjan valasztjuk ki a csapatot ami torlesre fog kerulni A fuggveny vegigiteral a csapatokon, és ha megtalalja a megadott nevvel rendelkezo csapatot, akkor torli azt a tarolobol. A torlés soran a csapat dinamikusan foglalt memoriateruletet felszabaditja, majd a tobbi csapatot balra tolja a tombben, hogy ne legyen res a torolt csapat utan. Vegul a tarolo meretet csokkenti.

4.5.2.2. get_csapat()

```
Csapat * Tarolo::get_csapat (
            size_t idx ) const  [inline]
```

Getter

Paraméterek

idx	- a tomb idx-edik elemet adjuk vissza Ha az index érvényes, akkor visszaadja az adott indexen lévõ csapatot a csapatok tömbbõl.

4.5.2.3. hozza_ad()

```
void Tarolo::hozza_ad (
            Csapat * cs )
```

Csapat hozzáadása a tárolóhoz

Paraméterek

cs	- hozzáadandó csapat pointer ha a meret megegyezik a kapacitassal akkor az egyenkenti hely foglalast elkerulve a (kapacitas * 2) helyet foglal a tarolonak

4.5.2.4. olvasas()

```
void Tarolo::olvasas (
            const String & filenev )
```

Tároló tartalmának beolvasasa egy filebol

Paraméterek

filenev	- file neve ahonnan a beolvasas tortenik Ha sikerult megnyitni a file-t akkor egy ciklus segitsegevel folyamatosan beolvassa a filban levo objektumokat, es eldonti a beiraskor bekerult szam alapjan hogy, melyik alosztaly fog kovetkezni

4.5.2.5. tarolo_fileba_iras()

```
void Tarolo::tarolo_fileba_iras (
            const String & filenev )
```

Tároló tartalmának kiírása fileba

Paraméterek

filenev	- file neve ahova a kiiras tortenik Ha sikerült megnyitni a fájlt, a függvény végigmegy a tárolóban található csapatokon egy ciklus segítségével. Minden iterációban meghívja az adott csapat, fileba iras metódusát Ezáltal az adott csapat adatai kiíródnak a fájlba

4.5.2.6. tarolo_kiir()

```
void Tarolo::tarolo_kiir (
            std::ostream & os ) const
```

Tároló tartalmának kiírása

Paraméterek

os	- kimeneti stream referencia Ha az elem nem üres, meghívja annak a kiir metódusát, amely az adott csapat adatait írja ki a megadott kimeneti streamre. Ezáltal minden csapat adatai kiíródnak

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- tarolo.h
- tarolo.cpp