Programozás alapjai 2.

Házifeladat Specifikáció

Steffler Máté u45x9

2024.04.07.

# Feladat:

Sportegyesület

A Fitt Sportegyesület nyilvántartást szeretne vezetni a csapatairól. Minden csapat rendelkezik egy névvel és egy alaplétszámmal. A sportegyesület háromféle sportággal foglalkozik: labdarúgás, kosárlabda és kézilabda. A labdarúgó csapatnak két edzője van; a kosárlabda csapatnak elengedhetetlen kellékei a pom-pom lányok aminek létszámát is nyilvántartják; a kézilabda csapatok pedig évente kapnak valamekkora összegű támogatást. A nyilvántartás rendelkezzen minimum az alábbi funkciókkal: új csapat felvétele, csapat törlése, listázás.

# Feladat specifikáció:

Fitt Sportegyesület nyilvántartását tartalmazó program megírása a feladat.

Program tárolja a sportegyesület csapatait melyek sportág szerint 3 kategóriában lehetnek. Minden kategóriában kötelezően tartalmazni kell egy csapatnak a nevét és a létszámát. A 3 kategória a labdarúgás a kosárlabda és a kézilabda. Plusz adatként labdarúgó csapat esetén két edző nevét is elkell tárolni, kosárlabda esetén a pom-pom lányok létszámát és kézilabda esetén az évente kapott támogatás összegét.

# Tesztelési dokumentáció:

1. Először a fájl beolvasást, egyesület létrehozást, és egyesülethez csapatok hozzáadását és ezek kilistázását teszteltem:

Ennek eredménye igy néz ki:

-----------------Csapatok listazasa-----------------

Csapat neve: Foci Csapat | Csapat Letszama: 11

Csapat neve: Kosar Csapat | Csapat Letszama: 20

Csapat neve: Uj kezi | Csapat Letszama: 8

Csapat neve: Kezi Csapat | Csapat Letszama: 8

Ezt úgy ellenőriztem, hogy a külső fájlban megadott adatokkal összevetetem és megegyezett.

1. Másodszorra egy csapatra kerestem rá és ennek minden adatát kiírtattam. Itt a keresés és a csapat összes adatának kiírását teszteltem.

Eredmény:

-----------------Kezi csapatra rakereses es kiras-----------------

Csapat neve: Kezi Csapat Csapat Letszama: 8

Sport neve: Kezi

PomPom letszam: 120

Ezt úgy ellenőriztem, hogy a külső fájlban megadott adatokkal összevetetem és megegyezett.

1. Harmadszorra az adat módosítás teszteltem ugyan ezen a csapaton és nevét, létszámát és PomPom létszámát modósítottam.

Eredmény:

-----------------Kezi csapatra rakeresas es atnevezes valamint adat modositas-----------------

Csapat neve: Regi kezi Csapat Letszama: 30

Sport neve: Kezi

PomPom letszam: 44

Itt összevetettem az általam megadott módosítássok és helyesen változtak az adatok

1. Csapat törlés teszteltem azzal, hogy töröltem azt a kézi csapatot, akikkel eddig dolgoztam, illetve itt a memória kezelést is néztem, hogy nem kapok-e hibát.

Eredmény:

-----------------Kezi csapat torlese es maradek kilistazasa-----------------

Sikeresen torolve a Regi kezi nevu csapatot

Csapat neve: Foci Csapat | Csapat Letszama: 11

Csapat neve: Kosar Csapat | Csapat Letszama: 20

Csapat neve: Uj kezi | Csapat Letszama: 8

Ellenőrizésnél látszik, hogy már nincs kilistázva.

1. Ötödiként azt vizsgáltam, hogy tudok e manuálisan csapatot felvenni és felvettem azt a csapatot, akit előbb töröltem csak az eredeti adataival.

Eredmény:

# -----------------Manualisan Kezi csapat felvetel -----------------

# Csapat neve: Foci Csapat | Csapat Letszama: 11

# Csapat neve: Kosar Csapat | Csapat Letszama: 20

# Csapat neve: Uj kezi | Csapat Letszama: 8

# Csapat neve: Kezi Csapat | Csapat Letszama: 8

Ellenőrzésként látszik, hogy bekerült a listába a csapat.

1. Fájlba írás úgy teszteltem, hogy visszatöltöttem a listát a fájlba és ha újra futtattjuk a programot látssza, hogy megint jól kéri be az adatokat.

# Programterv:

Program indításakor egy külső fájlból a program beolvassa az adatokat és aztán ezt eltárolja egy dinamikusan foglalt tömbe. Program innentől a tömbben lévő adatokkal dolgozik és mikor a programot bezárjuk akkor tömbbe lévő adatokat visszamenti a fájlba és következő indításkor innen tudja újra bemásolni.

## UML diagram:

A képen szöveg, nyugta, képernyőkép, Betűtípus látható

Automatikusan generált leírás

## Osztályok bemutatása:

**Egyesület:**

Az Egyesulet osztály egy sportegyesületet reprezentál, amely több csapatot tartalmaz.

Privát Tagok:

* int csapatszam - A csapatok száma az egyesületben.
* Csapat\* csapatok - A csapatok tömbje.

Publikus Tagok:

* Egyesulet()
  + Alapértelmezett konstruktor, amely nullázza a csapatok számát és a csapatok pointerét.
* void operator+(Csapat& ujCsapat)
  + Egy csapatot hozzáad a listához.
  + Paraméterek:
    - Csapat& ujCsapat - A hozzáadandó csapat.
* Csapat& keres(const char\* nev)
  + Keres egy adott nevű csapatot, és visszaadja annak referenciaját.
  + Paraméterek:
    - const char\* nev - A keresett csapat neve.
  + Visszatérési érték: Csapat& - A megtalált csapat referenciaja.
* int getCsapatszam() const
  + Visszaadja a csapatok számát az egyesületben.
  + Visszatérési érték: int - A csapatok száma.
* void operator-(const char\* nev)
  + Egy csapatot eltávolít a listából a neve alapján.
  + Paraméterek:
    - const char\* nev - Az eltávolítandó csapat neve.
* void list()
  + Kilistázza a csapatokat.
* void fajlIras()
  + Kiírja a csapatok adatait egy fájlba.
* void fajlBeolvasas()
  + Beolvassa a csapatok adatait egy fájlból.
* ~Egyesulet()
  + Elöszőr a csapatokhoz tarozó sportágakat törli aztán a csapatok tömböt.

**Csapat:**

A Csapat osztály egy sportcsapatot reprezentál, amely tartalmazza a csapat nevét, létszámát és a hozzá tartozó sportágat.

Privát Tagok:

* std::string csapatNev - A csapat neve.
* int csapatLetszam - A csapat létszáma.
* SportAg\* csapatSportAg - A csapathoz tartozó sportág (pointer a SportAg osztályra).

Publikus Tagok:

* Csapat()
  + Alapértelmezett konstruktor, amely üres implementációval rendelkezik.
* Csapat(const char\* csapatnev, const int csapatletszam, SportAg& sport)
  + Létrehoz egy Csapat objektumot char\* paraméterrel a csapat neve, létszáma és a sportág alapján.
  + Paraméterek:
    - const char\* csapatnev - A csapat neve.
    - const int csapatletszam - A csapat létszáma.
    - SportAg& sport - A csapathoz tartozó sportág.
* Csapat(std::string csapatnev, const int csapatletszam, SportAg& sport)
  + Létrehoz egy Csapat objektumot std::string paraméterrel a csapat neve, létszáma és a sportág alapján.
  + Paraméterek:
    - std::string csapatnev - A csapat neve.
    - const int csapatletszam - A csapat létszáma.
    - SportAg& sport - A csapathoz tartozó sportág.
* std::string getNev() const
  + Visszaadja a csapat nevét.
  + Visszatérési érték: std::string - A csapat neve.
* int getLetszam() const
  + Visszaadja a csapat létszámát.
  + Visszatérési érték: int - A csapat létszáma.
* SportAg\* getSportag() const
  + Visszaadja a csapathoz tartozó sportág pointerét.
  + Visszatérési érték: SportAg\* - A csapathoz tartozó sportág pointere.
* void csapatAdatKiiras()
  + Kiírja a csapathoz tartozó adatokat.
* void nevModosit(const char\* ujNev)
  + Módosítja a csapat nevét.
  + Paraméterek:
    - const char\* ujNev - Az új csapatnév.
* void letszamModosit(const int ujLetszam)
  + Módosítja a csapat létszámát.
  + Paraméterek:
    - const int ujLetszam - Az új létszám.
* ~Csapat()
  + Az osztály destruktora, amely alapértelmezett megvalósítással rendelkezik.
* std::ostream& operator<<(std::ostream& os, const Csapat& csap)
  + operátor túlterhelés az << operátor számára, amely lehetővé teszi a csapat adatainak kiírását egy kimeneti stream-re.
  + Paraméterek:
    - std::ostream& os - A kimeneti stream.
    - const Csapat& csap - A csapat objektum.
  + Visszatérési érték: std::ostream& - A kimeneti stream, amely lehetővé teszi a láncolást.

**SportAg:**

A SportAg osztály egy absztrakt bázisosztály, amely különböző sportágakat reprezentál.

Privát Tagok:

* std::string nev - A sportág megnevezése (pl. foci, röpi vagy kézi).

Publikus Tagok:

* SportAg(const char\* name)
* Létrehoz egy SportAg objektumot char\* paraméterrel.
  + Paraméterek:
    - const char\* name - A sportág neve.
* SportAg(std::string name)
  + Létrehoz egy SportAg objektumot std::string paraméterrel.
  + Paraméterek:
    - std::string name - A sportág neve.
* std::string getSportnev() const
  + Visszaadja a sportág nevét.
  + Visszatérési érték: std::string - A sportág neve.
* Virtuális Függvények
  + Ezek a függvények tisztán virtuálisak, és a leszármazott osztályokban felül kell őket írni.
  + virtual void nevModosit(const char\* ujnev, const char\* ujnev2) = 0
    - Módosítja az edzők nevét.
    - Paraméterek:
      * const char\* ujnev - Az első név.
      * const char\* ujnev2 - A második név .
  + virtual void adatModosit(const int ujszam) = 0
    - Módosítja a sportághoz kapcsolódó adatokat.
    - Paraméterek:
      * const int ujszam - Az új adat értéke.
  + virtual void adatKiiras() = 0
    - Kiírja a sportághoz kapcsolódó adatokat.
  + virtual void sportagAdatkiiras()
    - Kiírja a sportághoz kapcsolódó sport nevét.
  + virtual int getSzamAdat() = 0
    - Visszaad egy számadatot a sportághoz kapcsolódóan.
    - Visszatérési érték: int - Számadat.
  + virtual std::string getStringAdat() = 0
    - Visszaad egy string adatot a sportághoz kapcsolódóan.
    - Visszatérési érték: std::string - String adat.
  + virtual ~SportAg()

**Kosárlabda:**

A Kosarlabda osztály a SportAg osztály leszármazottja.

Privát Tagok:

* int pomLetszam - A pompomlányok létszáma.

Publikus Tagok:

* Kosarlabda(const char\* nev, int letszam)
  + Létrehoz egy Kosarlabda objektumot char\* paraméterrel a sport neve és a pompomlányok létszáma alapján.
  + Paraméterek:
    - const char\* nev - A sport neve.
    - int letszam - A pompomlányok létszáma.
* Kosarlabda(std::string nev, int letszam)
  + Létrehoz egy Kosarlabda objektumot std::string paraméterrel a sport neve és a pompomlányok létszáma alapján.
  + Paraméterek:
    - std::string nev - A sport neve.
    - int letszam - A pompomlányok létszáma.
* int getLetszam() const
  + Visszaadja a pompomlányok létszámát.
  + Visszatérési érték: int - A pompomlányok létszáma.
* void adatModosit(const int ujszam) override
  + Az adatModosit metódus felülírása, amely a pompomlányok létszámát módosítja.
  + Paraméterek:
    - const int ujszam - Az új létszám értéke.
* void nevModosit(const char\* ujnev, const char\* ujnev2) override
  + A nevModosit metódus felülírása, string adatot tudna módosítani.
* virtual void adatKiiras() override
  + Az adatKiiras metódus felülírása, amely pompomlányok létszámát írja ki.
* virtual int getSzamAdat() override
  + A getSzamAdat metódus felülírása, amely a pompomlányok létszámát adja vissza.
  + Visszatérési érték: int - A pompomlányok létszáma.
* virtual std::string getStringAdat() override
  + A getStringAdat metódus felülírása, amely string adatot tudna visszaadni.
* Destruktor
  + ~Kosarlabda()

**Kézilabda:**A Kezilabda osztály a SportAg osztály leszármazottja.

Privát Tagok:

* int tamogatas - Az éves támogatás összege.

Publikus Tagok:

* Kezilabda(const char\* nev, int evesTamogatas)
  + Létrehoz egy Kezilabda objektumot char\* paraméterrel a sport neve és az éves támogatás összege alapján.
  + Paraméterek:
    - const char\* nev - A sport neve.
    - int evesTamogatas - Az éves támogatás összege.
* Kezilabda(std::string n, int evesTamogatas)
  + Létrehoz egy Kezilabda objektumot std::string paraméterrel a sport neve és az éves támogatás összege alapján.
  + Paraméterek:
    - std::string n - A sport neve.
    - int evesTamogatas - Az éves támogatás összege.
* int getLetszam() const
  + Visszaadja az éves támogatás összegét.
* void adatModosit(const int ujszam) override
  + Az adatModosit metódus felülírása, amely az éves támogatás összegét módosítja.
  + Paraméterek:
    - const int ujszam - Az új támogatás összege.
* void nevModosit(const char\* ujnev, const char\* ujnev2) override
  + A nevModosit metódus felülírása, amely string adat típusokat tud módosítani.
* virtual void adatKiiras() override
  + Az adatKiiras metódus felülírása, amely a támogatást írja ki.
* virtual int getSzamAdat() override
  + A getSzamAdat metódus felülírása, amely az éves támogatás összegét adja vissza.
  + Visszatérési érték: int - Az éves támogatás összege.
* virtual std::string getStringAdat() override
  + A getStringAdat metódus felülírása, amely string adatot tudna visszaadni.
* ~Kezilabda()

**Labdarugás:**

A Labdarugas osztály a SportAg osztály leszármazottja.

Privát Tagok:

* std::string edzo1 - Az első edző neve.
* std::string edzo2 - A második edző neve.

Publikus tagok:

* Labdarugas(const char\* nev, const char\* edzok, const char\* edzok2)

Létrehoz egy Labdarugas objektumot char\* paraméterekkel

Paraméterek:

const char\* nev - A sport neve.

const char\* edzok - Az első edző neve.

const char\* edzok2 - A második edző neve.

* Labdarugas(std::string nev, std::string edzok, std::string edzok2)

Létrehoz egy Labdarugas objektumot std::string paraméterekkel.

Paraméterek:

std::string nev - A sport neve.

std::string edzok - Az első edző neve.

std::string edzok2 - A második edző neve.

* std::string getEdzo1() const

Visszaadja az első edző nevét.

* std::string getEdzo2() const

Visszaadja a második edző nevét.

* void adatModosit(const int ujszam) override

Az adatModosit metódus felülírása. Ha lenne int adat azt tudná modosítani.

Paraméterek:

const int ujszam - Az új adat értéke.

* void nevModosit(const char\* ujnev, const char\* ujnev2) override

A nevModosit metódus felülírása, amely az edzők nevét módosítja.

Paraméterek:

const char\* ujnev – edzo1 új neve

const char\* ujnev2 – edzo új neve

* virtual void adatKiiras() override

Az adatKiiras metódus felülírása, amely az edzők nevét írja ki.

* virtual int getSzamAdat() override

A getSzamAdat metódus felülírása, amely int adatokat tud kiirni.

* virtual std::string getStringAdat() override

A getStringAdat metódus felülírása, amely az edzők neveit adja vissza egyetlen stringben, pontosvesszővel elválasztva.

Visszatérési érték: std::string - Az edzők nevei pontosvesszővel elválasztva.

* ~Labdarugas()