

# Filmtár

## A választott feladat leírása

Készítsen filmeket nyilvántartó rendszert. minden filmnek tároljuk a címét, lejátszási idejét és kiadási évét. A családi filmek esetében korhatár is van, a dokumentumfilmek esetében egy szöveges leírást is tárolunk. Tervezzen könnyen bővíthető objektummodellt a feladathoz!

Demonstrálja a működést külön modulként fordított tesztprogrammal! A megoldáshoz ne használjon STL tárolót!

## Pontosított specifikáció

A filmtár egy filmeket nyilvántartó rendszer. Filmeket és azok adatait (cím, lejátszási idő<sup>1</sup>, kiadási év), továbbá annak műfaját (családi-, dokumentumfilm) is képes eltárolni.

A program lehetőséget biztosít cím szerinti keresésre az adatbázisból. Ezeken felül az adatbázisból kilistázhatóak a filmek műfajuk szerint is stb. Ezek és a további funkciók alább olvashatóak pontosabban.

## Menürendszer

A program felhasználói működtetését egy menürendszer vezérli. minden menüt egy-egy egész számmal jelöltünk. A program futtatása során egy ilyen egész számot fog várni a bemenetén.

<sup>1</sup> A lejátszási időt percben jeleníti meg a program, továbbá szintén ebben a mértékegységben várja a felhasználótól is a későbbi menüpontok során!

Abban az esetben, ha egy, a program számára nem ismert menüpontot adnak meg, akkor a kimeneten a(z) „**A MEGADOTT SZÁM NEM VOLT MEGFELELO**” szöveg látható. Ezt követően várja az új sorszámot.

Abban az esetben, ha a szám helyett szöveget adnak meg a programnak, akkor a „**HELYTELEN BEMENETEI ADATOK!**” szöveg jelenik meg a képernyőn.

### 1-es menüpont

**Kilistázza az adatbázisban szereplő összes film címét.** Soronként egy-egy film cím olvasható.

### 2-es menüpont

**Kilistázza az összes családi filmet.** Szintén soronként egy-egy adat lesz olvasható. Azonban az adatok a következők lesznek: cím, kiadás éve, lejátszási idő, korhatár.

### 3-as menüpont

**Kilistázza az összes dokumentumfilmet.** Az előző menüponthoz hasonlóan működik azzal a kivétellel, hogy a korhatár helyett egy rövid leírás lesz olvasható a filmről.

## 4-es menüpont

A program ekkor egy szöveget, pontosabban egy film címet vár bemenetként. A megadott adattal a program ezt követően egy keresést hajt végre. **A cím alapján megpróbálja megkeresni a filmet az adatbázisban.** Ha szerepel, akkor a film adatait fogja kiírni (címe, kiadási éve, lejátszási ideje és még valami<sup>2</sup>). Azonban, ha nem található, akkor a(z) „**ADATBÁZISBAN NEM SZEREPEL.**” szöveget írja ki.

## 5-ös menüpont

**Adatbázist bővíti.** Legelőször a program megkéri a felhasználót, hogy döntse el előre, hogy a film, amit meg fog adni, milyen műfajú (családi film/dokumentumfilm) lesz. Ezt követően a program három bemeneti adatot fog várni a felhasználótól, melyek sorban a következők lesznek: **film címe, kiadási éve, lejátszási ideje**.

Itt egy bonyolultabb ellenőrzésen fognak végig menni a beérkező adatok. (Cím során lehetőség van szám megadására is.) A cím megadásakor ellenőrzésre kerül, hogy a felhasználó egy olyat adott meg, ami már szerepel az adatbázisban. Ha már szerepel, akkor a „**ADATBÁZISBAN MÁR SZEREPEL!**” szöveget írja ki. Abban az esetben, ha a kiadási évnek vagy a lejátszási időnek szöveg típusú

adatot ad a felhasználó, a program a „**HELYTELEN BEMENETI ADATOK!**” figyelmeztető szöveget írja ki. Továbbá „**A MEGADOTT SZÁM NEM VOLT MEGFELELŐ!**” szöveg jelenik meg a kimeneten, ha a kiadási év kisebb, mint 1921 („A kölyök”) vagy nagyobb, mint az aktuális év.

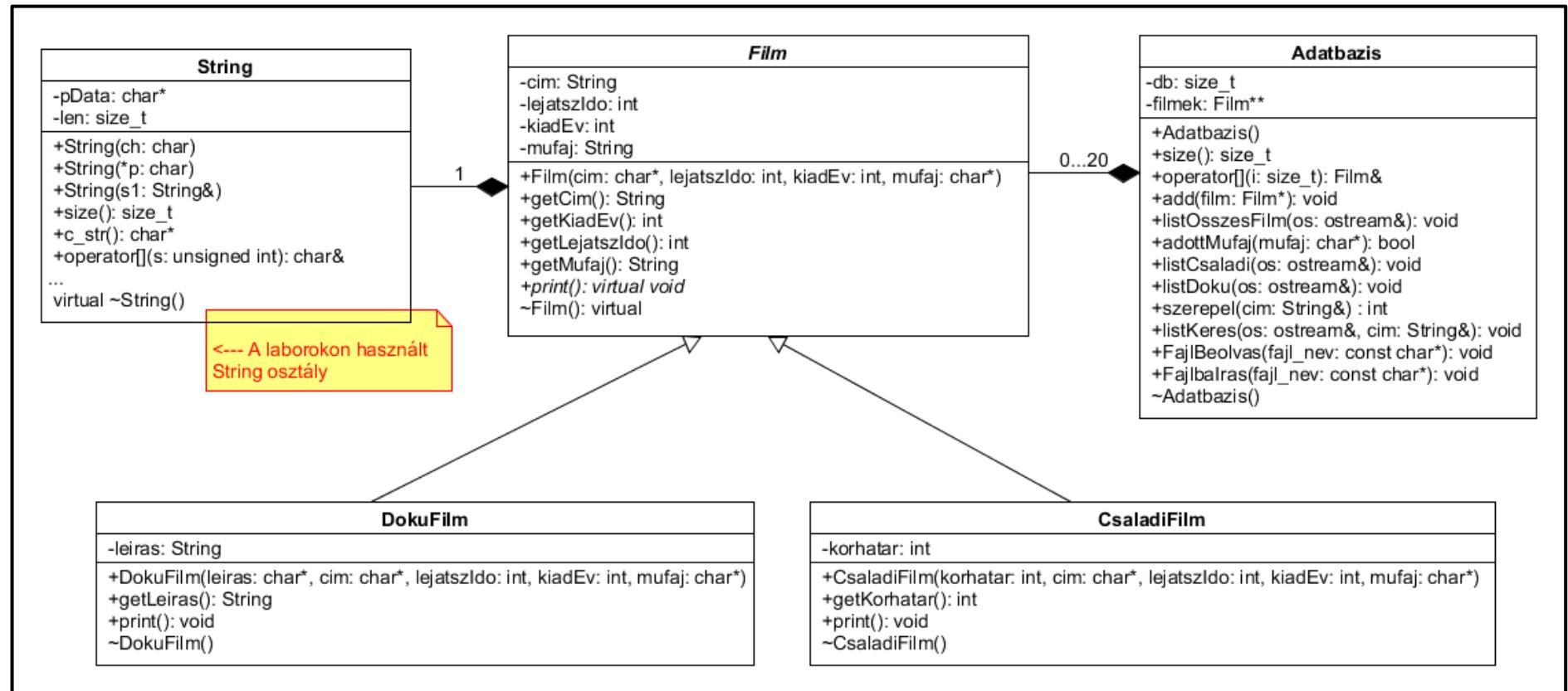
## 6-os menüpont

**Kilépés.** A program bezárja magát, ugyanakkor az élettartama során használt adatokat, tehát a filmeket, egy külső fájlba menti, ahonnan a program annak újraindításakor visszatölти azokat. Így lehetőség van az adatbázis bővítésére. A fájlban soronként a következő adatok találhatóak: „**műfaj:cím(kiadási\_év,lejátszási\_idő, korhatár/leírás)**”. Ha a memória megtelne a bővítések során, azt a „**MEMÓRIA MEGTELIK!!!**” szöveggel jelzi a kimeneten.

## Tesztesetek

Azon feltételezés útján, hogy a tesztesetek érdektelenek lesznek a felhasználók számára, ezért ezeket egy külön specifikált menüsorszámon keresztül lehet elérni. **A szám: 55.** A teszteseteket a specifikációban nem részletezzük, ezeket a dokumentációban tesszük közé.

<sup>2</sup> A film műfajától függően jelenik meg annak korhatára vagy leírása.

UML

1. ábra: UML diagramm

## Algoritmusok

Ebben az alrészben olvasható az a programozási téTEL, amely a feladat jelleGE miATT szükséGESSÉ válhat. KizáróLAG felsoroláskéNT írOM le, mivel minden jó programozó ismeri jóL.

Talán ITT szeretném azt IS megjegyezni, hogy habár az előadáson még nem volt szó absztrakt osztályok és tényleges dinamikus memória kezelésének kapcsolatáról, ugyanakkor sikerült a megvalósítása<sup>3</sup> (kigondolása). UtóbbIT az UML diagramm Adatbazis osztályában látható, a filmek nevezetű változónál.

- Lineáris keresés tétele (4-es menüpontban)

Itt a terv az, hogy egy keres (ld.: UML diagramm Adatbazis osztály) függvényvel hajtom végre a téTELt. Ha a megadott film cím szerepel az adatbázisban, akkor visszaadom annak indexét, ellenkező esetben *mínusz egyet*.

---

<sup>3</sup> Segítségre szolgált: [creating dynamic array of an abstract class](#)

# Film osztályreferencia abstract

Film ABSZTRAKT osztály. Részletek...

```
#include <film.h>
```

Leszármazottak: CsaladiFilm és DokuFilm.

## Publikus tagfüggvények

**Film** (const char \*cím, int lejatszIdo, int kiadEv, const char \*mufaj="N/A")  
Tagfüggvények. Részletek...

**String** **getCim** () const

Visszaadja a film címét. Részletek...

**int** **getKiadEv** () const

Visszaadja a film kiadási évét. Részletek...

**int** **getLejatszido** () const

Visszaadja a film lejátszási idejét. Részletek...

**String** **getMufaj** () const

Visszaadja a film műfaját. Részletek...

**virtual void** **print** (std::ostream &os) const =0

Kiirja a film attribútumait. Részletek...

**virtual** ~**Film** ()

Destruktor Megszünteti az objektumot.

## Privát attribútumok

**String** **cim**

Adattagok. Részletek...

**int** **lejatszido**

Film lejátszási ideje.

**int** **kiadEv**

Film kiadási éve.

**String** **mufaj**

Film műfaja.

## Részletes leírás

Film ABSZTRAKT osztály.

A cím-ben tároljuk a film címének karaktereit, a kiadEv a film kiadásának éve, a lejatszido a film lejátszási ideje PERCBEN, a mufaj a film műfajának a karaktereit tárolja.

Definíció a(z) film.h fájl 15. sorában.

# Konstruktörök és destruktörök dokumentációja

## ◆ Film()

```
Film::Film ( const char * cim,  
             int      lejatszido,  
             int      kiadEv,  
             const char * mufaj = "N/A"  
           )
```

inline

Tagfüggvények.

Paraméteres konstruktur. Ez a default is!

### Paraméterek

|            |                                    |
|------------|------------------------------------|
| cim        | - String típusú változó címnek     |
| kiadEv     | - int tip. vált. kiadási évnek     |
| lejatszido | - int tip. vált. lejátszási időnek |
| mufaj      | - String tip. vált. műfajnak       |

Definíció a(z) **film.h** fájl 33. sorában.

# Tagfüggvények dokumentációja

## ◆ getCim()

```
String Film::getCim ( ) const
```

inline

Visszaadja a film címét.

### Visszatérési érték

- String tip. változó.

Definíció a(z) **film.h** fájl 39. sorában.

## ◆ getKiadEv()

```
int Film::getKiadEv ( ) const
```

inline

Visszaadja a film kiadási évét.

#### **Visszatérési érték**

- int tip. vált.

Definíció a(z) **film.h** fájl 44. sorában.

### ◆ **getLejatszido()**

```
int Film::getLejatszido ( ) const
```

inline

Visszaadja a film lejátszási idejét.

#### **Visszatérési érték**

- int tip. vált.

Definíció a(z) **film.h** fájl 49. sorában.

### ◆ **getMufaj()**

```
String Film::getMufaj ( ) const
```

inline

Visszaadja a film műfaját.

#### **Visszatérési érték**

- String tip. vált.

Definíció a(z) **film.h** fájl 54. sorában.

### ◆ **print()**

```
virtual void Film::print ( std::ostream & os ) const
```

pure virtual

Kiirja a film attribútumait.

Tisztán virtuális függvény, emiatt ABSZTRAKT a **Film** osztály. A leszármazott osztályok **print()** függvényei hívódnak meg igazából.

Megvalósítják a következők: **DokuFilm** és **CsaladiFilm**.

## ◆ cím

String Film::cím

private

Adattagok.

Film cime

Definíció a(z) **film.h** fájl 18. sorában.

---

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- **film.h**
- 

Készítette **doxygen** 1.9.1

# CsaladiFilm osztályreferencia

Ősök: Film.

## Publikus tagfüggvények

**CsaladiFilm** (int korhatar, const char \*cim, int lejatszido, int kiadEv, const char \*mufaj="Csaladi film")

Tagfüggvények. Részletek...

int **getKorhatar** () const

Lekérdezi a film korhatárát.

void **print** (std::ostream &os) const

Kiirja a családi film adatait.

**~CsaladiFilm** ()

Destruktor Megszünteti az objektumot.

 Publikus tagfüggvények a(z) **Film** osztályból származnak

## Privát attribútumok

int **korhatar**

Adattag. Részletek...

## Részletes leírás

Definíció a(z) **csaladi\_film.h** fájl 14. sorában.

## Konstruktörök és destruktörök dokumentációja

◆ **CsaladiFilm()**

```
CsaladiFilm::CsaladiFilm ( int korhatar,
                           const char * cim,
                           int lejatszido,
                           int kiadEv,
                           const char * mufaj = "Csaladi film"
                           )
```

inline

Tagfüggvények.

Konstruktor Ez a default is!

#### Paraméterek

|            |                                    |
|------------|------------------------------------|
| cim        | - String típusú változó címnek     |
| kiadEv     | - int tip. vált. kiadási évnek     |
| lejatszido | - int tip. vált. lejátszási időnek |
| mufaj      | - String tip. vált. műfajnak       |
| korhatar   | - int tip. vált. korhatárnak       |

Definíció a(z) **csaladi\_film.h** fájl 30. sorában.

## Adattagok dokumentációja

### ◆ korhatar

```
int CsaladiFilm::korhatar
```

private

Adattag.

Családi film korhatára

Definíció a(z) **csaladi\_film.h** fájl 17. sorában.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- **csaladi\_film.h**

# DokuFilm osztályreferencia

Ősök: Film.

## Publikus tagfüggvények

**DokuFilm** (const char \*leiras, const char \*cim, int lejatszido, int kiadEv, const char \*mufaj="Dokumentumfilm")  
Tagfüggvények. Részletek...

**String getLeiras () const**  
Lekérdezi a film leirását.

**void print (std::ostream &os) const**  
Kiírja a dokumentumfilm adatait.

**-DokuFilm ()**  
Destruktor Megszünteti az objektumot.

 Publikus tagfüggvények a(z) Film osztályból származnak

## Privát attribútumok

**String leiras**  
Adattag. Részletek...

## Részletes leírás

Definíció a(z) **doku\_film.h** fájl 14. sorában.

## Konstruktörök és destruktörök dokumentációja

◆ **DokuFilm()**

```
DokuFilm::DokuFilm ( const char * leiras,  
                      const char * cim,  
                      int      lejatszido,  
                      int      kiadEv,  
                      const char * mufaj = "Dokumentumfilm"  
)
```

inline

Tagfüggvények.

Konstruktor Ez a default is!

#### Paraméterek

|            |                                    |
|------------|------------------------------------|
| cim        | - String típusú változó címnek     |
| kiadEv     | - int tip. vált. kiadási évnek     |
| lejatszido | - int tip. vált. lejátszási időnek |
| mufaj      | - String tip. vált. műfajnak       |
| leiras     | - String tip. vált. a leírásnak    |

Definíció a(z) **doku\_film.h** fájl **30.** sorában.

## Adattagok dokumentációja

### ◆ leiras

**String** DokuFilm::leiras

private

Adattag.

Dokumentumfilm leírása

Definíció a(z) **doku\_film.h** fájl **17.** sorában.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- **doku\_film.h**

# Adatbazis osztályreferencia

```
#include <adatbazis.h>
```

## Publikus tagfüggvények

**Adatbazis ()**

Tagfüggvények. Részletek...

**size\_t size () const**

**Film & operator[](size\_t i)**

**void add (Film \*)**

**void listOsszesFilm (std::ostream &os) const**

**bool adottMufaj (const char \*mufaj) const**

**void listCsaladi (std::ostream &os) const**

**void listDoku (std::ostream &os) const**

**int szerepel (String &p\_cim) const**

**void listKeres (std::ostream &os, String &p\_cim) const**

**~Adatbazis ()**

Adatbázis destruktora.

**void FajlBeolvas (const char \*fajl\_nev)**

**void Fajlbalras (const char \*fajl\_nev)**

## Részletes leírás

Adatbazis osztály. Itt tároljuk a filmeket egy dinamikus tömbben.

## Konstruktorok és destrukturerek dokumentációja

### ◆ Adatbazis()

Adatbazis::Adatbazis ( )

inline

Tagfüggvények.

Paraméter nélküli konstruktur. Üres adatbázis

## Tagfüggvények dokumentációja

### ◆ add()

```
void Adatbazis::add ( Film * film )
```

Film hozzáadása az adatbázishoz.

#### Paraméterek

**Film\*** - Absztrakt osztály tipusú változóra mutató pointer

Film hozzáadása az adatbázishoz

#### Paraméterek

**Film\*** - Absztrakt osztály tipusú változóra mutató pointer

### ◆ adottMufaj()

```
bool Adatbazis::adottMufaj ( const char * mufaj ) const
```

Kiértékeli, hogy egy adott műfajú film létezik-e az adatbázisban.

#### Paraméterek

**mufaj** - const char\* tipus a megadott műfajnak.

#### Visszatérési érték

- bool (TRUE, ha létezik. FALSE, ha nem.)

Kiértékeli, hogy egy adott műfajú film létezik-e az adatbázisban

#### Paraméterek

**mufaj** - const char\* tipus a megadott műfajnak

#### Visszatérési érték

- bool (TRUE, ha létezik. FALSE, ha nem.)

### ◆ Fajlbalras()

```
void Adatbazis::Fajlbalras ( const char * fajl_nev )
```

Fájlba ír filmeket egy megadott adatbázisból

#### Paraméterek

**fajl\_nev** - a fájl neve, amibe írjuk az adatokat

Fájlba ír filmeket az adatbázisból

#### Paraméterek

**fajl\_nev** - a fájl neve, amibe írjuk az adatokat

### ◆ FajlBeolvas()

```
void Adatbazis::FajlBeolvas ( const char * fajl_nev )
```

Kezeli a fájlból való beolvasást. Beolvassa a filmeket egy adatbázisba

#### Paraméterek

**ab** - az adatbázis, amibe olvasunk

**fajl\_nev** - a fájl, amiből olvasunk.

Kezeli a fájlból való beolvasást. Beolvassa a filmeket az adatbázisba

#### Paraméterek

**fajl\_nev** - a fájl, amiből olvasunk.

0: film cime 1: mufaja 2: kiadási éve 3: lejátszási ideje 4: mufaj szerinte: korhatár vagy leírás fajlban: "1:0(2,3,4)

Beolvasás

Mufaj

Cím

Kiadási év

Lejátszási idő

### ◆ listCsaladi()

```
void Adatbazis::listCsaladi ( std::ostream & os ) const
```

Kilistázza az összes olyan filmet, amelynek műfaja "csaladi film".

#### Paraméterek

**os** - ostream

Kilistázza az összes olyan filmet, amelynek mufaja "csaladi film"

#### Paraméterek

**os** - ostream

### ◆ listDoku()

```
void Adatbazis::listDoku ( std::ostream & os ) const
```

Kilistázza az összes olyan filmet, amelynek műfaja "dokumentumfilm".

#### Paraméterek

**os** - ostream

Kilistázza az összes olyan filmet, amelynek műfaja "dokumentumfilm"

#### Paraméterek

**os** - ostream

### ◆ listKeres()

```
void Adatbazis::listKeres ( std::ostream & os,  
                           String & p_cim  
                         ) const
```

Kezeli a **szerepel()** függvény visszatérési értékét.

#### Paraméterek

**cim** - const char\* a keresendo film cime.

**os** - ostream.

Kezeli a **szerepel()** függvény visszatérési értékét

#### Paraméterek

**cim** - const char\* a keresendo film cime

**os** - ostream

### ◆ listOsszesFilm()

```
void Adatbazis::listOsszesFilm ( std::ostream & os ) const
```

Kilistázza az összes film cimet az adatbázisban.

#### Paraméterek

**os** - ostream

Kilistázza az összes film cimet az adatbázisban

#### Paraméterek

**os** - ostream

### ◆ operator[]( )

## Film & Adatbazis::operator[] ( size\_t i )

Adott film lekérdezése az adatbázisból. Nem létező 'i' index esetén kivételt dob.

### Paraméterek

i - az index maga - size\_t típusú!

### Visszatérési érték

- Ha helyes és létező indexet kapott, akkor visszaadja a tároló i-edik elem címét, ellenkező esetben ez egy új elemre való hivatkozás, így az új elem címét adja majd vissza.

Adott film lekérdezése az adatbázisból Nem létező 'i' index esetén kivételt dob

### Paraméterek

i - az index maga - size\_t típusú!

### Visszatérési érték

- Ha helyes és létező indexet kapott, akkor visszaadja a tároló i-edik elem címét, ellenkező esetben ez egy új elemre való hivatkozás, így az új elem címét adja majd vissza

## ◆ size()

size\_t Adatbazis::size ( ) const

inline

Tároló méretének lekérdezése.

### Visszatérési érték

- size\_t típ. vált. a méretnek

## ◆ Szerepel()

int Adatbazis::szerepel ( String & p\_cim ) const

Film cimet keres az adatbázisban.

### Paraméterek

cim - const char\* a keresendo film cime.

### Visszatérési érték

int - a (megtalált) film indexe az adatbázisban. Ha nem szerepelt benne, akkor minusz egyet ad.

Film cimet keres az adatbázisban

### Paraméterek

cim - const char\* a keresendo film cime

### Visszatérési érték

int - a (megtalált) film indexe az adatbázisban. Ha nem szerepelt benne, akkor minusz egyet ad.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- C:/Users/User/Iskola/!BME/!Szemeszter2/Programozás/Házi feladat/Program/filmtar/adatbazis.h
- C:/Users/User/Iskola/!BME/!Szemeszter2/Programozás/Házi feladat/Program/filmtar/adatbazis.cpp

## Tesztesetek

A tesztesetek száma tíz, melyek közül minden arra hívott, hogy ellenőrizze az osztályok kapcsolatát és az osztályok tagfüggvényeinek helyességét. A következőkben ezekről lesz szó bővebben.

### *családi film.print*

Teszteljük az adatok kiíratását: cím, kiadási év, lejátszási idő és műfajtól függően – ami most „családi film” – a korhatár. A teszt lényege, hogy belássuk, hogyha egy családi film adatait szeretnénk kiíratni, akkor a korhatár jelenjen meg, ne más.

### *dokumentumfilm.print*

Az előzőhez hasonlóan teszteljük a film adatainak kiíratását, csak most műfajtól függően a korhatár helyett a film leírását szeretnénk megjeleníteni.

### *adatbázis.konstruktur*

A teszt előtt létrehozzunk egy adatbázist, tehát meghívjuk az Adatbazis osztály konstruktörét, ami nullával inicializálja a filmeket tartalmazó dinamikus tömböt. Magyarán, létrehoz egy olyan dinamikus tárolót, amelynek mérete egyenlő a nullával.

Ez a teszteset azt vizsgálja, hogy tényleg nulla kezdőmérettel jön-e lére az objektum.

### *adatbázis.add*

Hozzádunk egy filmet az adatbázishoz, majd ellenőrizzük, hogy tényleg bekerült-e úgy, hogy kiírjuk az (előbb létrehozott) adatbázis összes filmjét, ami ennél a tesztesetnél még csak 1-nek kell lennie. Kiírva az adatbázis összes filmének a címét, pontosan egy darabot, és azt az egy darabot kell látnunk, amit korábban a tesztesetnél hozzáadtunk.

### *adatbázis.listOsszesFilm*

Ebben a tesztesetben kettő új filmet adunk a korábban létrehozott adatbázishoz, vagyis jelenleg három film található benne. Ezekre meghívjuk a listOsszesFilm() Adatbazis osztály tagfüggvényét, amellyel kilistázzák az összes film címét az adatbázisban. A kimeneten pontosan azt a három címet kell kapnunk, amiket a korábbiakban pontosan hozzáadtunk.

### *adatbázis.listKeres*

Most megkeressünk egy adott filmet cím szerint. A teszt azt vizsgálja, hogyha egy létező film címét adunk meg a tagfüggvénynek, akkor visszaadja-e (kiírja-e a kimenetre) a film összes adatát.

### *adatbázis.listKeres(nincs)*

Megpróbálkozzunk azzal is, ha olyan film címre keresünk rá, ami az adatbázisban nem létezik. Ekkor a specifikációban leírtak szerint hibaüzenet kell küldeni a felhasználónak. Azt teszteljük tehát, hogy a megfelelő hibaüzenetet kapjuk-e vissza.

### *adatbázis.operator[]*

Ellenőrizzük a túl indexelést. Jelenleg a korábbiakat követve három film szerepel az adatbázisban, tehát a következő elemhivatkozáskor a tagfüggvénynek kivételt kell dobnia: *film[3]*. Azt teszteljük tehát, hogy a megfelelő hibaüzenetet kapjuk-e vissza.

### *adatbázis.szerepel()*

Film keresése üres adatbázisban. A teszesetben létrehozzunk egy új üres adatbázist, amiben keresést próbálunk végrehajtani. Mivel a gondolatmenet adja magát, ilyen nem lehetséges, illetve ez a művelet teljesen felesleges így. A tagfüggvénynek mírusz egyet kell visszatérítenie.

Ugyanígy megpróbálkozunk azzal is, ha nem üres adatbázisban egy nem létező filmre keresünk rá, ekkor a tagfüggvénynek szintén mírusz egyet kell visszatérítenie.

### *adatbázis.listCsaladi()*

Végül megpróbálkozunk a csak „csaladi film” műfajú filmek kilistázásával. A korábban létrehozott adatbázisban, aminek elemszáma három, hajtjuk végre a keresést, melyben egy családi film volt. Azt teszteljük, hogy a kimeneten a családi film adatai jelennek-e meg.

Hasonló elven működne a listDokufilm() tagfüggvény is, ahol viszont a dokumentumfilmek kerülnek terítékre.