Dokumentáció a 7-es beadandó feladathoz

Pataki Péter (M0YUME) OEP 22.csoport 2025.05.05.

7. Modellezzük egy **filmszínház** jegyvásárlásait!

A filmszínház több vetítőteremben vetíti a filmeket (előadások) megadott időintervallumokban. A vetítőtermek ülőhelyeinek száma eltérő: a széksorokat betűk (A, B, C, …), az székeket számok (1, 2, 3, …) azonosítják. A jegyek vásárlása előtt lehetőség van azok lefoglalására. A foglalt jegyeknél nyilvántartjuk a vásárlót (aki egy potenciális néző), aki egy egyedi azonosító megadásával tudja majd igazolni magát a jegy kifizetésekor. Egy jegy árát a vetített film határozza meg, amit módosít a vetőterem fajtájától (nagy, közepes, kis, VIP), és a jegyet felhasználó néző státuszától (felnőtt, diák, gyerek, nyugdíjas, törzstag) függő kedvezmény:

jegyár = alapár · Szorzó(terem, néző)

kedvezmény	gyerek	diák	felnőtt	nyugdíjas	törzstag
kis terem	40	20	0	30	40
közepes terem	30	30	5	20	40
nagy terem	20	40	10	20	40
VIP terem	0	0	5	0	0

a. Lehessen új vetítő termet létrehozni, egy régit megszűntetni.
b. Lehessen filmvetítést végezni: ehhez filmet kell beszerezni, előadást szervezni az egyik terem adott időpontjában.
c. Lehessen egy nézőnek törzstaggá válni.
d. Lehessen egy nézőnek jegyet foglalni, azt visszamondani, vagy megvenni
e. Melyik filmet nézte meg a legtöbb néző?
f. Számoljuk meg egy adott előadásra megvett, csak lefoglalt, illetve szabad helyeket!

Készítsen használati eset diagramot! Adjon meg a fenti feladathoz egy olyan objektum diagramot, amely megmutatja egy filmszínház egy vetítőtermét, ahhoz rendelt két előadást, azt, hogy egyik előadásra két helyet már egy néző lefoglalt, egy harmadikat egy másik néző már kifizetett.

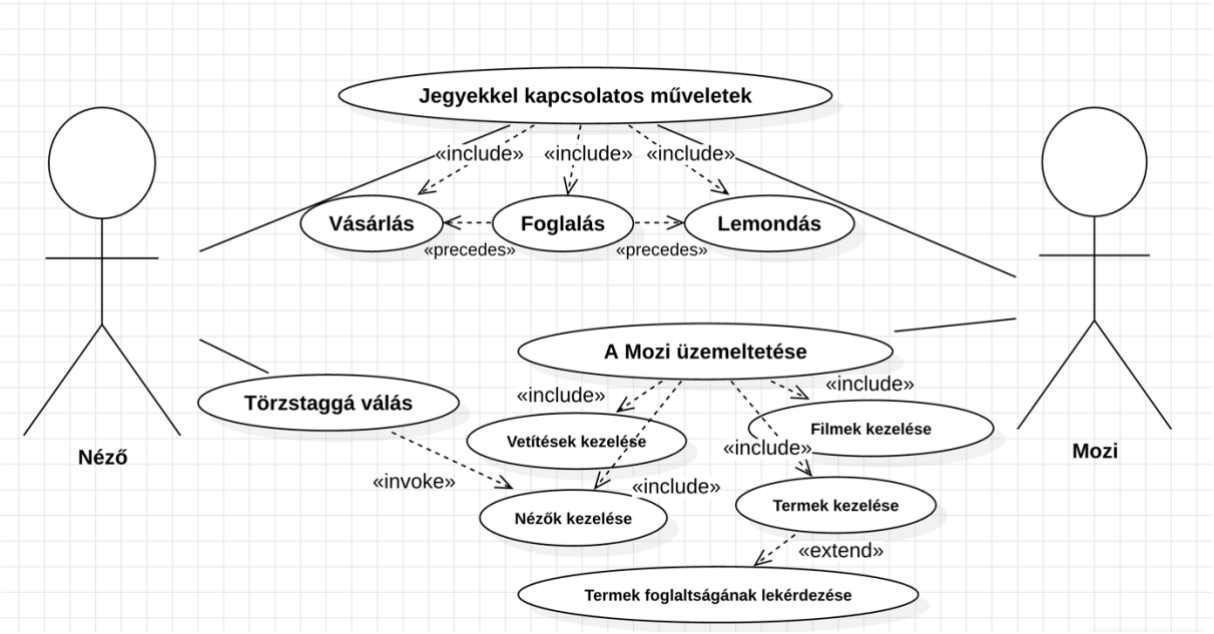
Készítse el egy jegy objektum állapotgépét! Minden ülőhelyhez tartozik egy jegy, amely lehet „szabad", „foglalt", vagy „eladott". Tervezze meg az állapot-átmeneteket megvalósító tevékenységeket, amelyeket majd az Jegy osztály metódusaiként definiálhat. Egészítse ezt ki kommunikációs diagrammá!

Rajzolja fel a feladat osztály diagramját! Felteheti, hogy a rejtett adattagokhoz mindig tartozik egy publikus getter: ha mégsem, akkor azt a „secret" megjegyzéssel jelölje. Egészítse ki az osztálydiagramot az objektum-kapcsolatokat létrehozó metódusokkal, valamint a feladat kérdéseit megválaszoló metódusokkal. A metódusok leírása legyen minél tömörebb (például ciklusok helyett a megfelelő algoritmus minta specifikációs jelölését használja). Használjon tervezési mintákat, és mutasson rá, hogy hol melyiket alkalmazta.

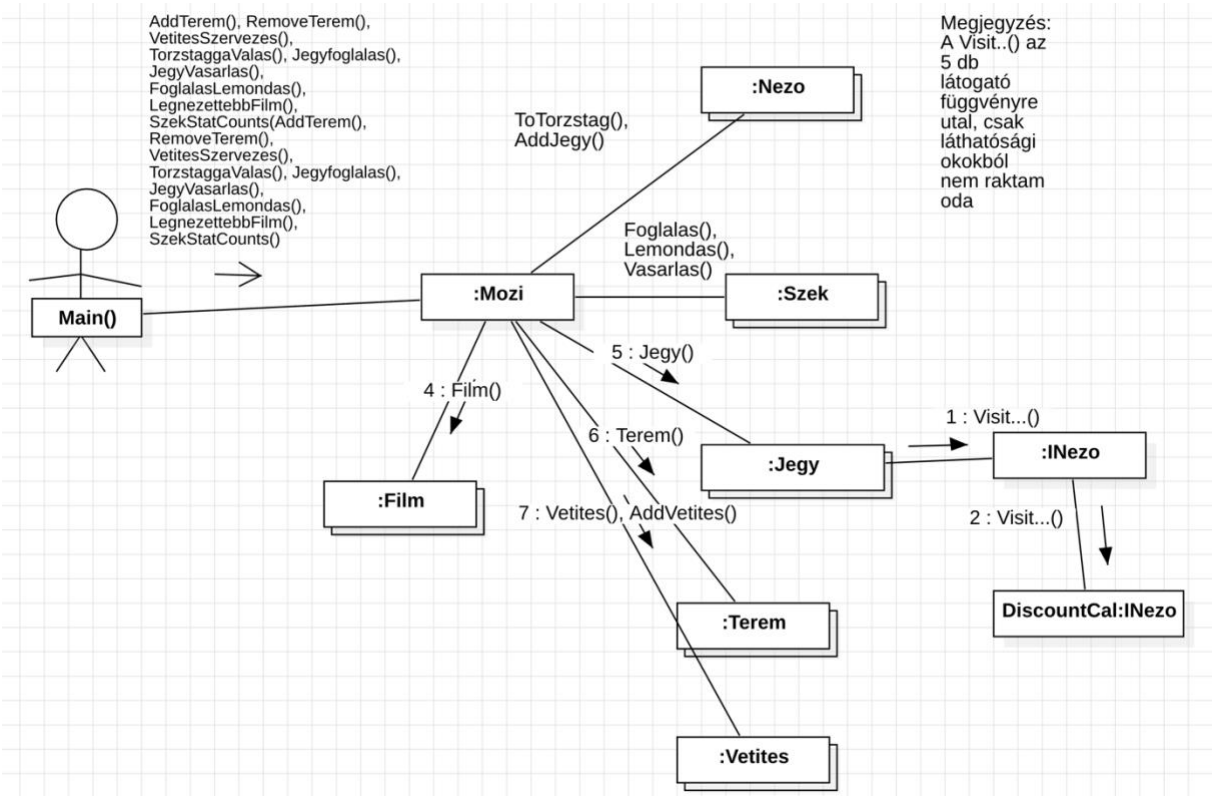
Implementálja a modellt! Szerkesszen olyan szöveges állományt, amelyből fel lehet populálni egy filmszínházat vetítőtermekkel és filmekkel. Meg lehessen hirdetni előadásokat (adott filmre, adott teremben), amelyre a nézők jegyeket foglalhatnak, foglalást visszamondhatnak, illetve jegyeket vásárolhatnak (akár foglalás nélkül is). Válaszolja meg az e. és f. kérdéseket. Készítsen teszteseteket, néhánynak rajzolja fel a szekvencia diagramját, és hozzon létre ezek kipróbálására automatikusan tesztkörnyezetet!

Terv

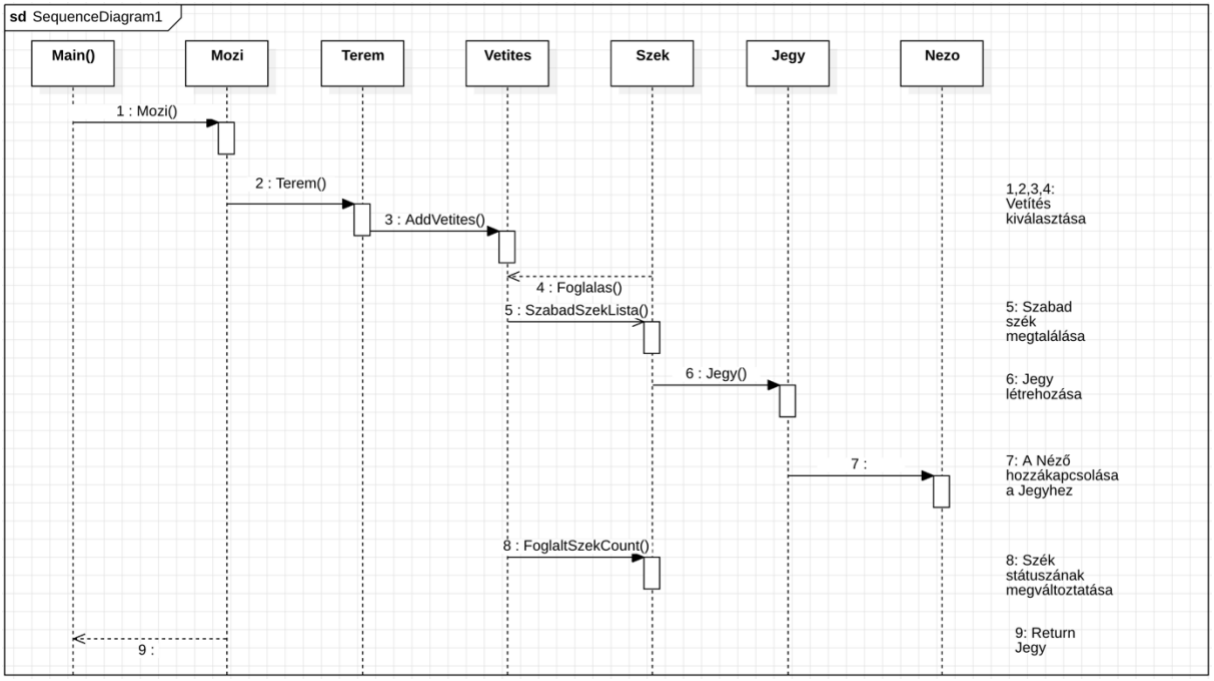
Ha a néző oldaláról tekintjük a program funkcionalitását, akkor egy egyszerű két részből álló feladatunk van. A néző vehet foglalhat, lemondhat és vásárolhat jegyet, valamint törzstaggá válhat. A mozi kezelője, ezeket az igényeket elégíti ki. Mivel azonban egy jegy, rengeteg információt tartalmaz (vetítés, terem, a konkrét szék a teremben, a néző típusa, és a jegy ára) ezért a mozi oldaláról több használati eset lehetséges (pl. Legnézettebb film lekérdezése extra funkció).



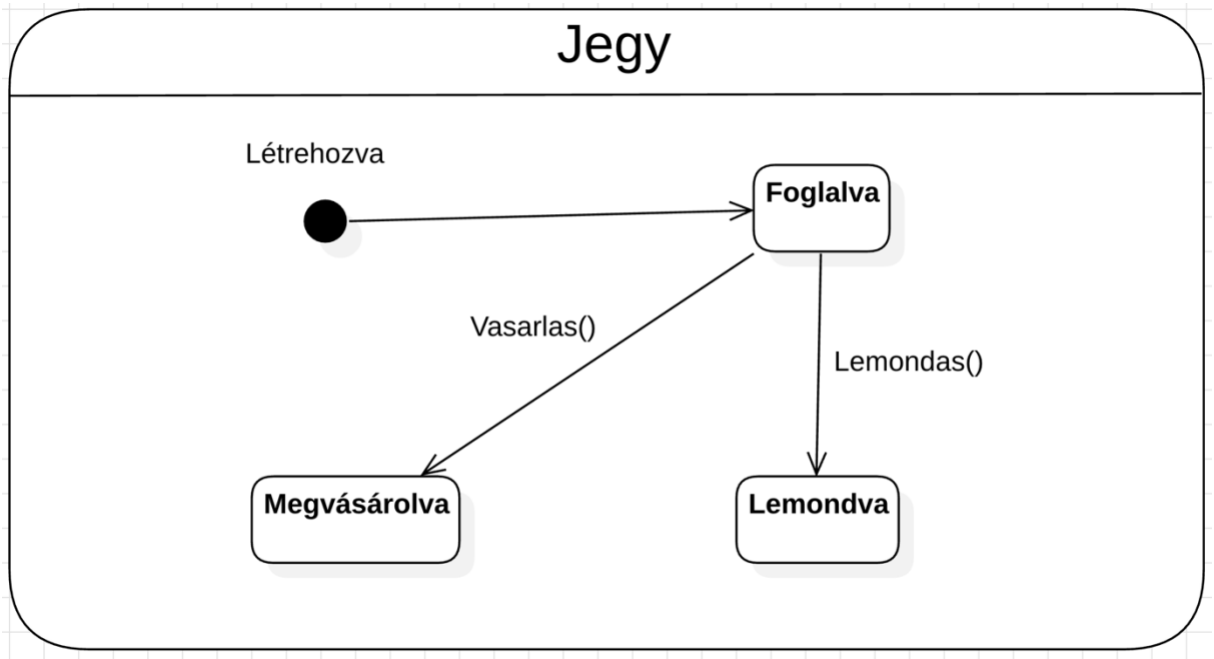
Kulcsfontosságú objektum a *Mozi,* ami egységesíti a funkcionalitást, és hívja a többi objektum függvényeit.



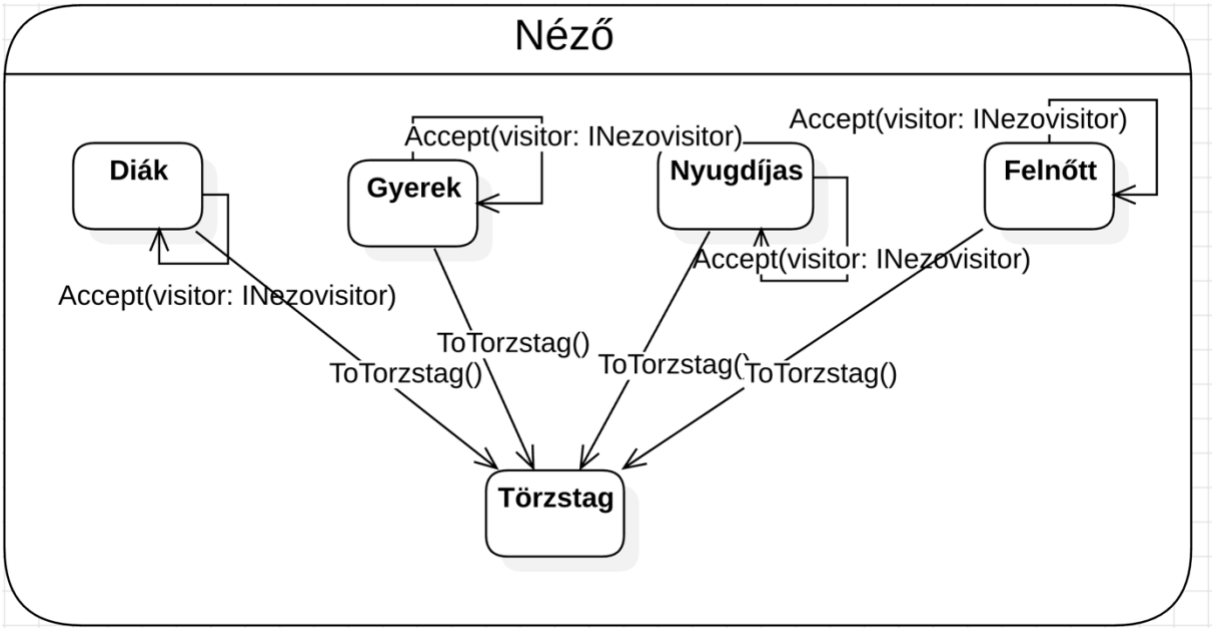
Az alábbi Szekvencia diagramban egy *Néző* jegyfoglalását követhetjük végig. A *Main* program pédányosít egy M*ozi* objektumot majd az hívja a megfelelő függvényeket. A procedúra két ellenőrzést is tartalmaz. Egyszer a *Foglalas()* függvény meghívása után ellenőrizzük, hogy a szabad *Székek* között van a lefoglalni kívánt ülőhely, majd amikor az adott *Nézőhöz* hozzákapcsoltuk a létrehozott *Jegyünket* megnézzük, hogy az adott *Vetítés* adott *Termében* tényleg változott e a *Székek* állapota.



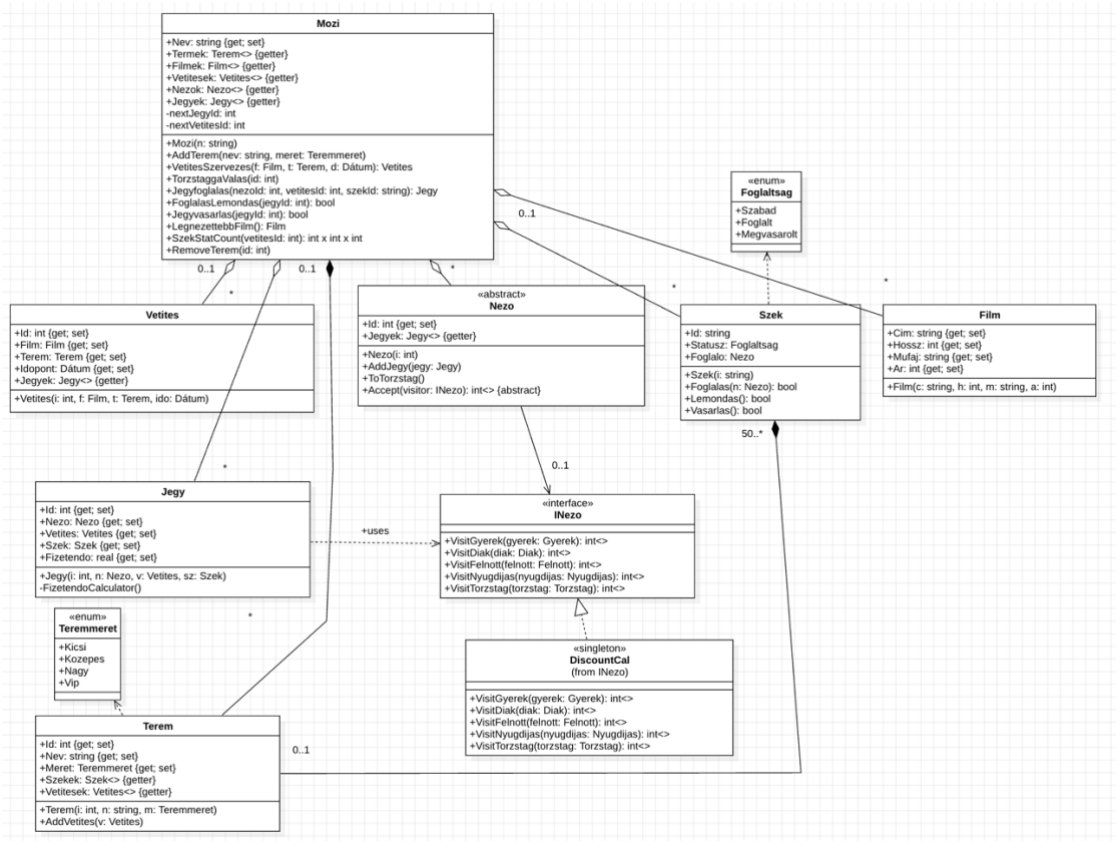
A *Jegy* objektum *állapotgépe* bemutatja ahogy a konstruktor meghívása után pédányosodik egy *Jegy* és utána milyen tranzíciókat végezhet. Érdemes megfigyelni, a létrehozott *Jegy* objektum kizárólag *„Foglalva”* állapotba mehet át, tehát nem lehet egyből vásárolni. Ilyenkor egy *Néző* objektumhoz rendelődik hozzá a *Jegy*. Ez azért van így tervezve hogy átláthatóbb legyen a program.



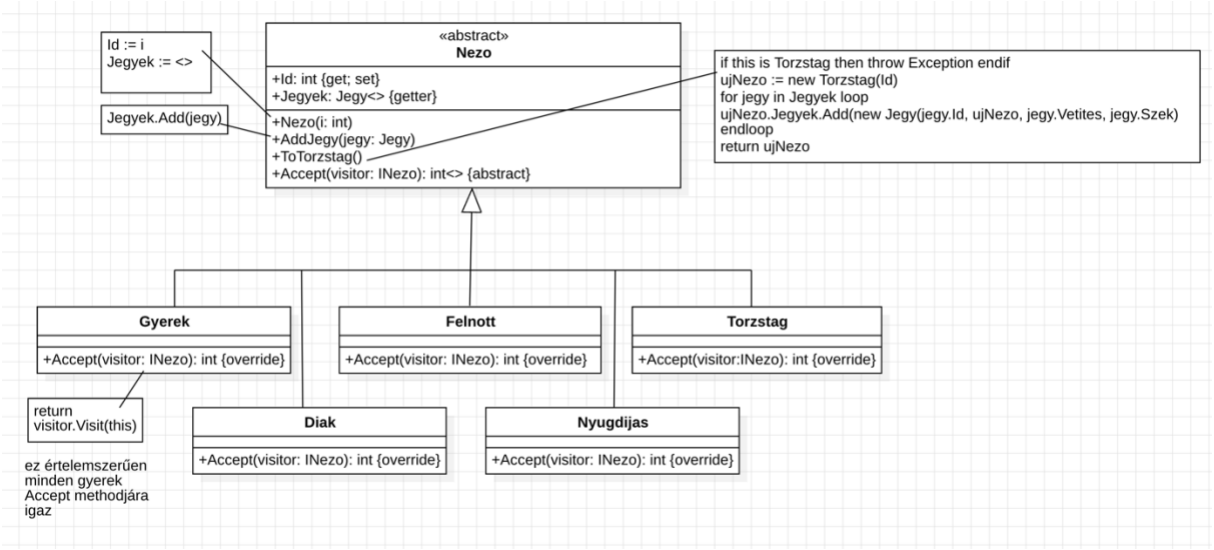
A Néző egy absztrakt osztály aminek gyerekei vannak, aszerint hogy a Mozi nyílvántartása, milyen kedvezményre jogosító kategóriába sorolja őket. Ezeket a gyerekeket látogatja egy másik objektum, különböző return értékekkel. (Az értékek 0-100-ig terjedő, százalékokat reprezentáló számok).



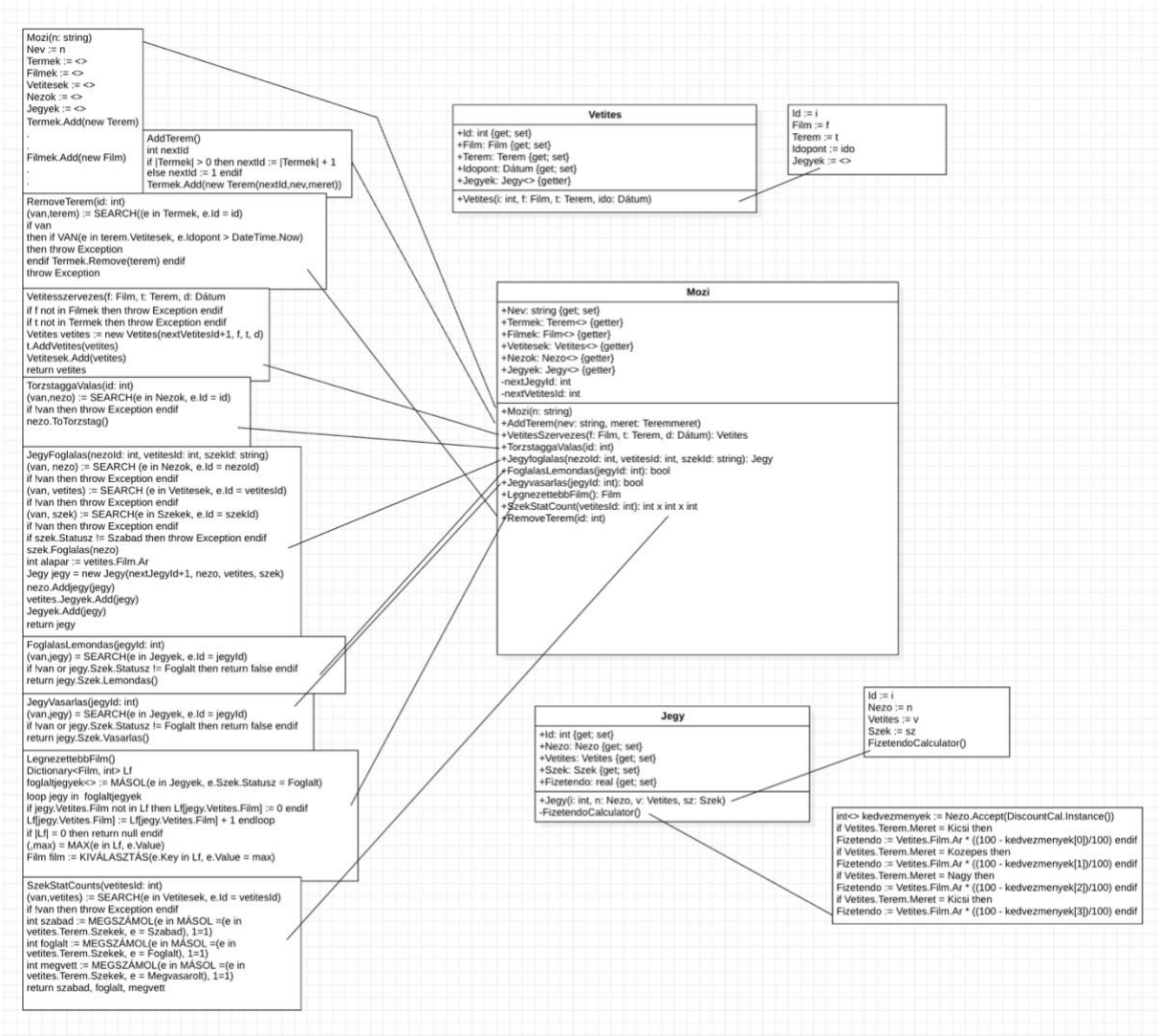
A következő *Osztálydiagram* részleten majdnem az összes *osztály* rajta van (kivéve a *Néző* gyerekei) ebből látszik, hogy ez a feladat akkor implementálható jól ha sok kisebb összefonódó *osztályt* csinálunk (pl. A *Termeken* belül *Szék* objektumok vannak), így elkerülve hogy fedjék egymást az *osztályok* felelősségkörei ami ennek a programozási paradigmának a legfontosabb alapelve.

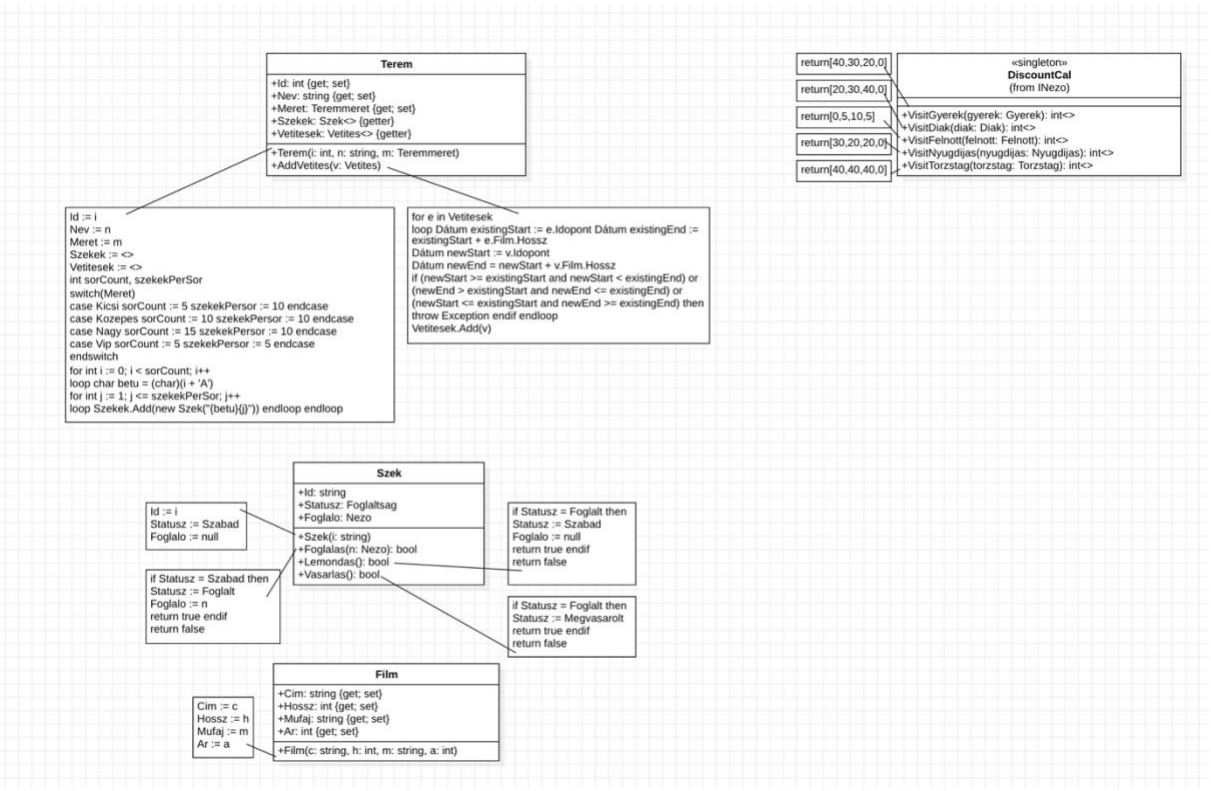


Az osztálydiagram *Látogató* és *Egyke* tervezési mintát tartalmaz



A következő képeken a többi függvénytörzs látható, egy ilyen komplex feladatnál is rengeteg alapvető programozási tételt lehet használni, ez segít egységesíteni a tervet és biztonságosabbá, hatékonyabbá teszi a Metódusainkat





Tesztelési Terv

A Main() függvény és a Unit tesztek megírása során a következőket kell tesztelni, hogy a program funkcionalitása teljes mértékben kielégítse a feladat leírását:

- Minden alfeladat „a-f”, működik, különös figyelmet fordítunk a program életszerűségére (pl. ne lehessen két előadást szervezni ugyanabban az időpontban ugyanabba a terembe vagy egy jegyet csak akkor lehessen visszamondani ha már tényleg le is foglalták)
- A feladat leírásában lévő táblázat lehető összes eshetőségét tesztelni kell, hogy helyesen kalkulál-e a programunk végösszeget a vásárláshoz.

Megjegyzések: A Mozi osztály függvénytörzseinek első sorában szerepel a függvény szignatúrája is ha esetleg követhetetlen lenne a nyilazás. Az Objektum Diagram nem szerepel a tárgy hivatalos honlapján lévő mintadokumentációban sem, ezért nem tette bele a dokumentumba.