5. feladat

A turisták látogatása bevételt hoz egy városnak, de kis mértékben rontja is a város állapotát. Egy város, ami jó állapotban van, vonzza a turistákat. Egy rossz állapotú város taszítja az odalátogatni készülőket.

Egy turista látogatása átlagosan 100.000 Ft bevételt hoz a városnak. Ha a város bevétele egy évben meghaladja az egy milliárd forintot, az egy milliárdon felüli részt a város javítására és szépítésére fordítják, hogy több látogató érkezzen a következő évben. A város állapota 1 és 100 pont között mozog (1 alá és 100 fölé sose megy, mert az állam elkölti a fölösleget és besegít, ha már nagyon vészes a helyzet). 1 és 33 közt számít lepusztultnak, 34 és 67 között átlagosnak és 67 fölött jó állapotúnak. Minden évben egy milliárd forint bevétel fölött minden húszmillió forint hoz egy pont állapotjavulást a városnak.

A turisták 3 fajtába sorolhatók: a japánok rendet raknak maguk után, így ők nem rontják a város állapotát. A modern országokból érkező turisták kevésbé ügyelnek a környezetükre: 100-asával rontanak egy-egy pontot a város állapotán. A harmadik csoportba sorolható turisták azon országok képviselői, ahol a szemetelés kulturális szokásnak tekinthető, ők 50-esével rontanak egy-egy pontot a város állapotán.

Ha a város jó állapotban van, abban az évben 20%-kal több japánt és 30%-kal több modernt vonz, mint ahány tervezte, hogy ellátogat oda. Átlagos állapotban 10%-kal több modernt és 10%-kal több harmadik típusú turistát vonz. Lepusztult állapot esetén a japánok egyáltalán nem jönnek, a többiek pedig annyian, amennyien tervezték.

**Adjuk meg, hogy a fájlban jelölt évek letelte után milyen a város állapota! Körönként mutassuk meg az érkezett turisták számát (hány tervezett és hány jött) kategóriák szerint, az éves bevételt és a város felújítás előtti állapotát (szám és kategória)!**

A program egy szövegfájlból olvassa be az adatokat! Az első sorban a város kezdeti állapota szerepel. A második sor jelöli a szimulált évek számát. A következő sorok tartalmazzák, hogy az egyes években hány turista tervezte, hogy eljön a városba: minden sor 3 darabszámot tartalmaz (japánok, modernek, többiek). A program kérje be a fájl nevét, majd jelenítse is meg a tartalmát. (Feltehetjük, hogy a fájl formátuma helyes.)

1. Elemzés

A ténylegesen városba látogatók számát a tervezetthez képest az alábbi táblázat szemlélteti a város állapotának és a látógató típusának függvényében:

|  |  |  |  |
| --- | --- | --- | --- |
| Turista/ Állapot | Japán | Modern | Egyéb |
| Jó | 120% | 130% | 100% |
| Közepes | 100% | 110% | 110% |
| Lepusztult | 0% | 100% | 100% |

A város állapota a pontok függvényében:

|  |  |  |  |
| --- | --- | --- | --- |
| Pont | 1-33 | 34-67 | 68-100 |
| Állapot | Lepusztult | Közepes | Jó |

A turisták pusztításának mértéke típusuk és lélekszámuk függvényében:

|  |  |  |  |
| --- | --- | --- | --- |
| Pusztít | Japán | Modern | Egyéb |
| Kritikus lélekszám | 1 | 100 | 50 |
| Mérték | 0 | 1 | 1 |

1. Terv

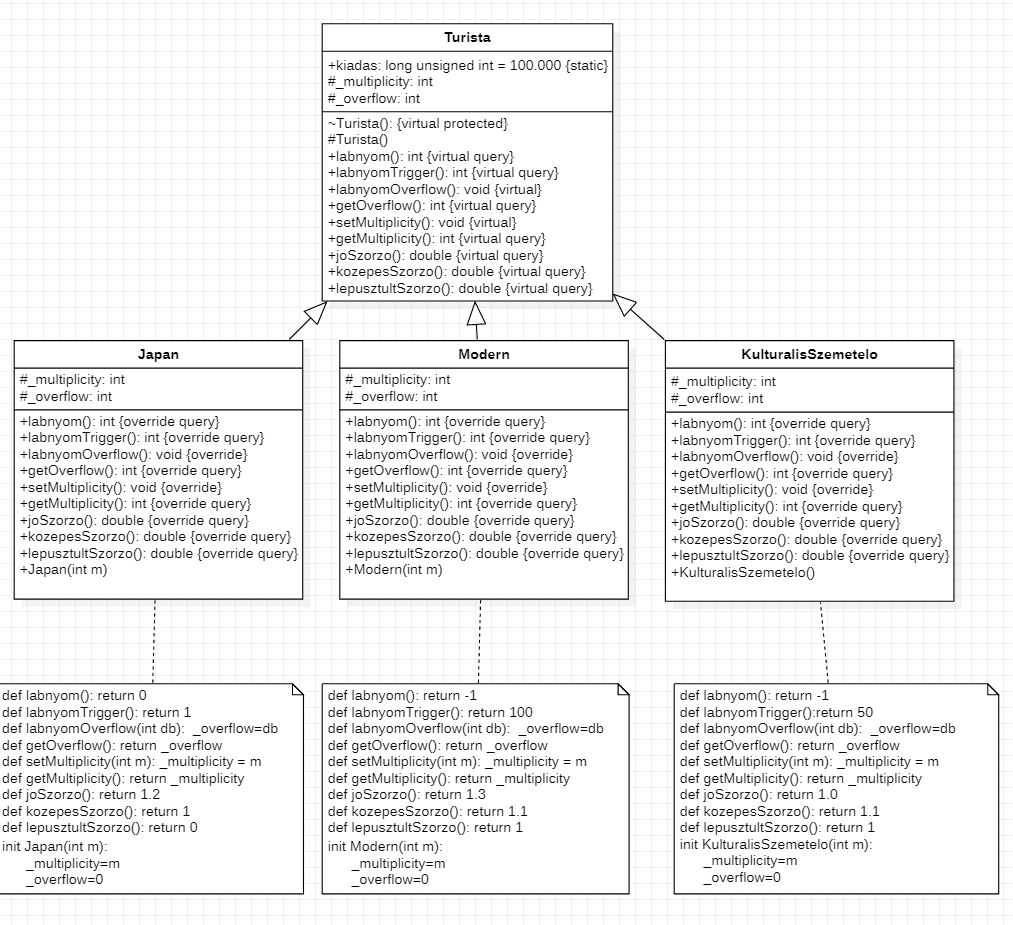
A kompozíció a kevésbé szigorú szabályok alapján van implementálva, lévén, hogy egykéket tartalmaz.

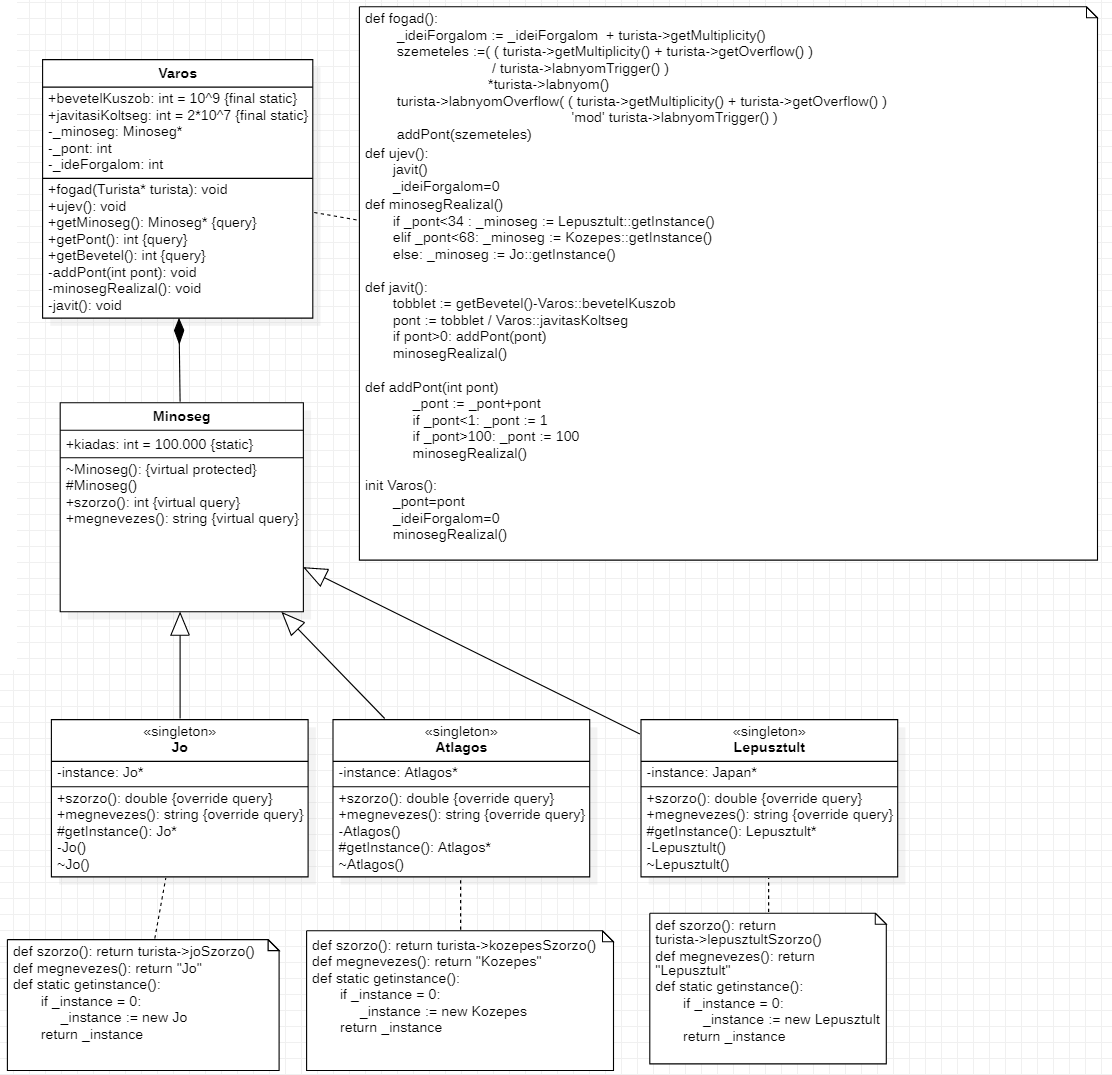
A turisták azért nem egykék, annak ellenére, hogy itt csak egy-egy lesz belőlük, mert azzal például nem tudnánk egyszerre több városon kiszámolni a problémát egy többszálas operáció keretében.

A turisták ha 50-esével rontanak, akkor nem hanyagolom el évek között a maradékokat. A bevételek viszont csak az adott évre vonatkoznak.

Megtalálható tervmintákra példák: Minőség alosztályai az egyke tervminta szerint vannak megvalósítva. A Turista virtuális metódusai, amit a konkrét turistaosztályok implementálnak, a sablonfüggvény tervmintát valósítja meg. A Minőség osztályok szorzó függvényei a kapott pointert annak dinamikus típusa szerinti metódusát hívja egy adott érték kinyeréséhez, ez a stratégia tervminta megvalósítása.

Diagramok a következő oldalakon





Egy évben érkező különböző fajtájú és számú turisták hatása a városra nézve. A város állapota önmagában jellemzi a rontás és a forgalom értékeket, amit az osztálydiagramban specifikáltam, ezért itt már nem kell jellemezni.

megj: egy nem objektumelvű megoldás valahogy így nézne ki:

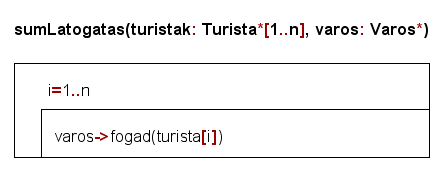
Állapottér: (turisták: (Turista\*)n)  
Ef: (turisták=turisták’)  
Uf: ( )

**Új állapotteret vezetünk be:**

Állapottér: (turisták: (Turista\*)n, Város\* v)  
Ef: (turisták=turisták’, v=v’)  
Uf: ( )//A turista objektumok állapota is változik, de egymástól függetlenül, ezért indexelhető itt a turisták tömb.

Visszavezetés: öszegzés

enor(E) ~ 1..n = 1..|turisták|  
f(e) ~ fogad(turisták[i]) //képzeljük azt hogy a város visszatér a „this”-el, azaz a saját új állapotával, de ezt a gyakorlatban implementálni teljesen felesleges  
s ~ város  
H,+,0 ~ Város\*, ⊜, város



Az összes év kiírása szintén egy összegzésre vezethető vissza, ahol párhuzamosan szimuláljuk illetve kiírjuk a város állapotát. Az összegzésben további összegzések vannak beágyazva.

Állapottér: (f: infile(Turistan), g: outfile(Szöveg), Város\* v, turisták: (Turista\*)n, evek: N, m: Minőség\*)  
Ef: (v=v’, f=f’)  
Uf: ( , )

Visszavezetés: öszegzés – külső ciklus

enor(E) ~ 1..n = 1..|evek|  
f(e) ~ (város->getMinoseg()->megnevezes(), város->getPont(), város->getBevetel())  
s ~ város  
H,+,0 ~ Szöveg, ⨁, <>

Visszavezetés: öszegzés -- A tervezettek kiírása

enor(E) ~ 1..n = 1..|turisták|  
f(e) ~ (turistak[i]->getMultiplicity(), i) //i a típusát egyértelműsítő érték ami kiírásra kerül  
s ~ város  
H,+,0 ~ N, ⨁, <>

Visszavezetés: öszegzés -- alias:

enor(E) ~ 1..n = 1..|turisták|  
f(e) ~ new Turista ( turistak[i]->getMultiplicity() \* m->szorzo(turistak[i]) )  
s ~ turistak’’  
H,+,0 ~ Turista\*, ⨁, <>

Visszavezetés: öszegzés -- A tervezettek kiírása

enor(E) ~ 1..n = 1..|turisták|  
f(e) ~ (turistak[i]->getMultiplicity(), i)  
s ~ város  
H,+,0 ~ N, ⨁, <>

