[Személyszállítás](https://lengyel.web.elte.hu/node/64" \l "comment-245) dokumentáció

Programozási Technológia 1.

Első beadandó

Kuti Bence Gilbert

HS74YU

**A feladat**

Személyszállítás: ország, város, jármű, busz, vonat, ember, vezető, ellenőr, utas.  
Létrehoz egy országot, benne egy-több várost, járműveket, majd a bevételt méri.  
Az emberek utaznak, ehhez jegyet vagy bérletet kell venniük.  
Ha nem vesznek jegyet, bérletet, és ellenőrrel találkoznak, büntetést fizetnek.

**Tervezés**

A feladat főbb célja egy személyszállítási modell létrehozása és szimulálása volt. Egy adott országon belül lévő városhoz/városokhoz tartozó vonat, busz forgalom szimulálása. A járműveken utazó vezető és ellenőr reprezentálása továbbá egy utasokért felelős osztály megtervezése amelynek rendelkeznie kell jeggyel vagy bérlettel. Ha nincs neki az ellenőr megbünteti. A büntetés és a jegy, bérlet vásárlás bevételt generál ezt is tárolni kellett.

**Megvalósítás**

A feladat alapján szükségünk van több osztályra:

**Ország:** felelős az ország eltárolásáért (láncolt listában) és az országhoz tartozó városok hozzáadásáért és eltávolításáért ezekhez egy Varos típusú listát használ a program. Ezekért ezen osztályon belül az addVaros() és a removeVaros felel().

**Város:** felelős a város eltárolásáért (láncolt listában) az országhoz tartozó bevétel hozzáadásáért addBevetel(), az országhoz tartozó aktuális bevétel lekérdezéséért actualBevetel(), megállapítja a Tartozik\_e\_orszaghoz() függvény segítségével, hogy van-e a városhoz tartozó ország. Továbbá felelős az országhoz való jármű hozzáadásáért és törléséért: addJarmu() és removeJarmu() függvények segítségével.

**Járművek:** egy egyszerű toString() metódusa egy switch-case szerkezettel eldönti, hogy milyen járművet vettünk fel éppen.

**Jármű:** a járművek reprezentációja felelős azért, hogy a jártmű hány férőhelyes, tartozik-e városhoz: addTartozik\_e\_varoshoz() és ennek az állapotának lekéréséért: Tartozik\_e\_varoshoz() továbbá képes az adott járműhez hozzárendelni és eltávolítani vezetőt: addVezeto(), removeVezeto() függvények segítségével. Utast hozzáadni addUtas() vizsgálni, hogy az adott utasnak van-e jegye, bérlete és ha nincs megbünteti és a büntetés értékével növeli a bevételt vagy utast leszállítani removeUtas(). Ellenőrt hozzáadni és leszállítani a járműről addEllenőr() és removeEllenőr() függvények segítségével.

**Busz**: a jármű childja amiben overrideolja, felüldefiniálja az addVezeto() metódust amely megnézi, hogy utazik-e a sofőr (felszállt-e a buszra), tartozik-e városhoz és ellenőrzi, hogy van-e jogosítványa a buszhoz.

**Vonat:** a jármű childja amiben overrideolja, felüldefiniálja az addVezeto() metódust amely megnézi, hogy utazik-e a sofőr (felszállt-e a vonatra), tartozik-e városhoz és ellenőrzi, hogy van-e jogosítványa a vonathoz.

**Ember:** felelős az ember névvel való felruházásáért és a nevének lekérdezéséért getNev().

**Vezető:** az ember childja amiben kiterjeszti a Vezetőt és felruházza azzal, hogy neki van jogosítványa. az addJogosítvány() függvénnyel hozzá is rendelhetünk jogosítványt.

**Ellenőr:** az ember osztályt terjeszti ki. Nevet ad az ellenőrnek.

**Utas:** az utas osztály reprezentálja a felszálló adott utast aki tud jegyet vásárolni jegyVasarlas(), bérletet vásárolni berletVasarlas(). Továbbá megnézi, hogy az utasnak van e jegye Van\_e\_jegy().

**Személyszállítás:** a fent felsorolt osztályok működését szimulálja és vizsgálja egy példaprogram segítségével.

Az összes osztály a személyszállítás csomagba került.