Szkriptnyelvek - Python ismertető

- A programot Python nyelven kell megírni.
- A benyújtandó fájl neve: `feladat.py`
    - Egy Python nyelven írt, szöveges fájl (nem zip, rar, stb.)
    - Ez csak a feladatban kért dolgokat tartalmazza! Amennyiben saját inputtal teszteled a kódot lokálisan, úgy feltöltés előtt a tesztelő kódrészletet kommenteld ki!
- A megoldást Bíró2 webes felületén (https://biro2.inf.u-szeged.hu) keresztül kell benyújtani és a megoldást a Bíró fogja kiértékelni.
    - A Feladat beadása felületen a *Feltöltés* gomb megnyomása után ki kell várni, amíg lefut a kiértékelés. **Kiértékelés közben nem szabad az oldalt frissíteni vagy a Feltöltés gombot újból megnyomni** különben feltöltési lehetőség veszik el!
- Feltöltés után a Bíró a programot **Python 3.11.1** interpreterrel fogja futtatni, és különböző tesztesetekre futtatja.
- A program működése akkor helyes, ha a tesztesetek futása nem tart tovább 2 másodpercnél és hiba nélkül fejeződik be, valamint a program működése a feladatkiírásnak megfelelő.
- Ha 3 teszteset futási ideje túllépi a fenti időkorlátot, a tesztelés befejeződik, a pontszám az addig szerzett pontszám lesz.
- A riport.txt megtekinthető az alábbi módon:
    1. Az Eredmények megtekintése felületen a vizsgálandó próba új lapon való megnyitása
    2. A kapott url formátuma: https://biro2.inf.u-szeged.hu/Hallg/IB370G/FELADAT_SZAMA/hXXXXXX/4/riport.txt
    3. Az url-ből visszatörölve a 4-esig (riport.txt törlése) megkaphatók a 4-es próbálkozás adatai
- A programot 25 alkalommal lehet benyújtani, a megadott határidőig.
- A munkád során figyelj arra, hogy pontosan kövesd a feladatban leírtakat, az elnevezéseket!
- A fájl elejére kommentbe írd be a neved, Neptun és h-s azonosítód az alábbi formában:

```
# Nev: Vezeteknev Keresztnev
# Neptun: NEP4LF
# h: h123456
```

Palackok

A feladatunk egy palackokkal foglalkozó alkalmazás alapjainak elkészítése.
Ehhez három osztályra lesz szükségünk: `Palack`, `VisszavalthatoPalack`, `Rekesz`.
Az alábbiakban az egyes osztályok lesznek bemutatva.
Fontos, hogy az osztályokat ebben a sorrendben készítsd el, mivel a tesztelés során számítunk rá, hogy a korábbi osztály(ok) alapjaiban léteznek (adattagok, konstruktorok megfelelőek).

Palack (34 pont)

A `Palack` osztály reprezentálja az általános palackokat az alkalmazásunkban.
A palackokról az alábbi adatokat tároljuk:

- a benne lévő ital (`ital` néven)
- maximális űrtartalom milliliterben (`max_urtartalom` néven)
- jelenleg benne lévő ital mennyisége milliliterben (`_jelenlegi_urtartalom` néven)

Készíts **inicializáló függvényt**, ami a fenti sorrendben várja az adatokat. A `jelenlegi űrtartalom` legyen opcionális, ha nem adjuk meg, akkor 1 legyen az értéke. A függvény a paraméterek alapján állítsa be az adattagokat.

Készíts **propertyt** a `_jelenlegi_urtartalom` adattaghoz, `jelenlegi_urtartalom` néven. A getter adja vissza az adattag értékét, a setter pedig az alábbiak szerint működjön:

- amennyiben a beállítani kívánt űrtartalom több, mint a palack űrtartalma, úgy a `_jelenlegi_urtartalom` értéket a palack maximális űrtartalmára állítsa be.
- minden más esetben állítsa be a jelenlegi űrtartalmat a paraméterből érkezőre.
- amennyiben 0-ra állítjuk be a jelenleg benne lévő ital mennyiségét, úgy az `ital` adattag értékét állítsa be a függvény `None`-ra.

Készítsd el a `suly` függvényt, ami nem vár paramétert. A függvény térjen vissza a palack súlyával. A palack súlya: `maximális űrtartalom/35 + jelenleg benne lévő ital súlya` (ami megegyezik az űrtartalommal).

Készítsd el a ***szöveggé alakító függvényt***, ami térjen vissza az alábbi formátumú szöveggel:

```
"Palack, benne levo ital: {ital neve}, jelenleg {jelenlegi űrtartalom} ml van
benne, maximum {max űrtartalom} ml fer bele."
```

Készítsd el az ***egyenlőségvizsgáló függvényt*** is. Két palack akkor egyenlő, ha mindegyik adattaguk megegyezik.

Készítsd el a **+= operátort megvalósító függvényt** is (`__iadd__`). Ezzel egy palack tartalmát lehessen hozzáadni az aktuális palackhoz. Ez a függvény a következőképp működjön:

- A jelenlegi űrtartalomhoz adja hozzá a másik palackban található ital jelenlegi űrtartalmát. Természetesen ne engedje, hogy többet töltsünk bele, mint amennyi belefér az adott palackba (ehhez használd a már elkészített property függvényt).
- Amennyiben a két `ital` megegyezik (egy narancsléhez narancslevelt adunk), akkor az operátor ne csináljon semmit az `ital` adattaggal.
- Amennyiben a két `ital` nem egyezik meg, az elkészült finomság neve legyen `keverek`, abban az esetben, ha mindkét palackban volt valamilyen ital. Amennyiben a jelenlegi palack üres volt, a másik palackban lévő ital neve kerüljön az `ital` adattagba. Feltehetjük, hogy üres palackból sosem szeretnénk beleönteni a saját palackunkba (tehát a paraméterben lévő palack `ital` adattag sosem None).

Visszaváltható palack (13 pont)

Készítsd el a `VisszavalthatoPalack` osztályt, ami egy speciális palackfajta.

A visszaváltható palackokról az alábbi adatot tároljuk a palack adatain felül:

- palackdíj Forintban (`palackdij` néven)

Készíts **inicializáló függvényt**, ami az ősosztály inicializálásához szükséges paraméterek után várja a palackdíjat. A palackdíj szintén opcionális, ha nem adjuk meg, akkor legyen 25 Ft. A függvény állítsa be az adattagokat (az ősosztály megfelelő metódusának meghívásával, valamint a palackdíj beállításával).

Készítsd el a ***szöveggé alakító függvényt***, ami használja fel a Palack azonos függvényét, azonban írja be elé, hogy "Visszavalthato", a megfelelő formátum tehát:

```
"visszavalthatoPalack, benne levo ital: {ital neve}, jelenleg {jelenlegi
űrtartalom} ml van benne, maximum {max űrtartalom} ml fer bele."
```

Rekesz (18 pont)

Készítsd el a `Rekesz` osztályt. A rekeszekről az alábbi adatokat tároljuk:

- maximális teherbírás (`max_teherbiras` néven)
- a tárolt palackokat (`palackok` néven)

Készíts **inicializáló függvényt**, ami csak a maximális teherbírást várja, a `palackok` adattagot pedig egy üres listával inicializálja.

Készítsd el a `suly` nevű függvényt, ami nem vár paramétert. A függvény térjen vissza a rekeszben tárolt palackok összsúlyával. Amennyiben nincs benne palack, térjen vissza 0-val.

Készítsd el a `uj_palack` nevű függvényt, ami egy palackot vár paraméterként. A függvény számolja ki, hogy ha beletennénk a paraméterben érkező palackot a rekeszbe, akkor azt elbírná-e a rekesz. Amennyiben nem bírná el, ne csináljunk semmit; ellenkező esetben pedig tegyük be a palackok közé a paraméterben érkező palackot.

Készítsd el a `osszes_penz` nevű függvényt, ami nem vár paramétert. A függvény számolja ki, hogy mennyi pénzt kapunk, ha visszaváltjuk a rekeszben lévő összes palackot (tehát adjuk össze a palackdíjakat). Ügyeljünk rá, hogy a sima `Palack` példányoknak nincs `palackdij` adattagjuk!

Jó munkát!