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://biro.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 3.12** 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://biro.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
```

Szkriptnyelvek 1. ZH

1. feladat: Palindrom idő (10 pont)

Készítsd el a `palindrom_ido` függvényt, amely egy időpontot vár szöveges formátumban (`<óra>:<perc>`).
A függvény célja annak meghatározása, hogy az időpont palindrom alakú-e. (Azaz, az óra karakterről karakterre megfordítva megegyezik a perccel.)

A függvény az alábbi lépések alapján működik:

- Ellenőrizd, hogy a bemenetként kapott érték szöveg-e. Ha nem, a függvény térjen vissza `False` értékkel. Amennyiben szöveg volt a bemenet, az garantáltan `<óra>:<perc>` formátumú.
- Győződj meg arról, hogy az óra értéke 0 és 23 közötti, míg a perc 0 és 59 közötti értéket vesz fel. Amennyiben bármelyik feltétel nem igaz, a függvény térjen vissza hamissal.
- Előfordulhat, hogy az óra egy számjegyű, ebben az esetben egészítsd ki az órát egy `0` karakterrel. A perc minden esetben két számjegyből áll.
- A függvény adjon vissza igaz értéket, ha az időpont palindrom alakú, és `False`-t egyébként.

Példa:

```
Input: "12:21"
Output: True  # Mert "12:21" visszafordítva is "12:21"

Input: "08:80"
Output: False  # Mert a perc érvénytelen

Input: "14:59"
Output: False  # Mert érvénytelen

Input: "3:30"
Output: True  # Mert "03:30" visszafordítva is "03:30"
```

2. feladat: Fesztivál (30 pont)

Készítsd el a `Fesztival` nevű osztályt, amely egy rendezvény helyszínét és résztvevőit kezeli.

Az osztálynak a következő adattagjai vannak:

- `nev`: a fesztivál neve, alapértelmezett értéke `"nincs"`.
- `ferohely`: a fesztivál teljes befogadóképessége, alapértelmezett értéke `1000`.
- `latogatok`: a jelenlegi látogatók száma (bármely objektum létrehozásakor `0`).
- `biztonsagi_orok`: egy lista, amely a biztonsági őrök nevét tárolja, ez bármely objektum létrehozásakor kezdetben egy üres lista.

Készítsd el az osztály konstruktorát mely a nevet és a férőhelyet várja.

`biztonsagi_felvetel` metódus

A metódus egy új lehetséges biztonsági őr (jelentkező) adatait fogadja el dictionary formájában, amelynek legalább egy mezője van: `nev`. Ezeken felül a jelentkező képességeit is megkapjuk, például: `cicasimogatas`. A metódus működése a következőképp zajlik:

- Ha a jelentkező nem rendelkezik megfelelő képességgel (`orzes_vedes` kulcs nem létezik, vagy az értéke `False`), akkor nem tudjuk felvenni a biztonsági őrt, a metódus térjen vissza `False`-szal.
- Ha van hely a fesztiválon (azaz a látogatók és biztonsági őrök száma nem haladja meg a fesztivál férőhelyét), adja hozzá a jelentkező nevét a `biztonsagi_orok` listához, és térjen vissza igazzal metódus.
- Ha nincs hely a fesztiválon, dobjon egy kivételt `"Nincs hely"` üzenettel.

`biztonsag_rendben` metódus

Ez a metódus nem fogad paramétert. A fesztivál biztonsága akkor van rendben, ha minden 20 látogatóra jut legalább egy biztonsági őr. Ebben az esetben a metódus térjen vissza igazzal. Ha nem jut minden 20 főre biztonsági őr, a metódus térjen vissza hamissal.

Tipp: 20 fő alatt nincs szükség biztonsági őrre, 39 fő esetén csak egy biztonsági őrre van szükség, 40 látogató esetén 2 őrre van szükség (egészosztás?!).

Hozzáadás (`__iadd__`) metódus

Implementáld a `+=` operátort, amely két `Fesztival` típusú objektumot egyesít.

- Ha a paraméterben érkező objektum nem `Fesztival` típusú, a metódus nem csináljon semmit, és térjen vissza az eredeti objektummal.
- Egyébként egyesítse a két fesztivál látogatóinak számát és biztonsági őreinek listáját, valamint növelje a férőhelyet az összeadott értékkel.
- Az módosított eredeti fesztivál objektumot adja vissza az operátor.

`__str__` metódus

Valósítsd meg az objektum szöveges reprezentációját biztosító `__str__` metódust, az alábbi formátumú szöveggel térjen vissza:

```
"A {nev} fesztivalon jelenleg {latogatok} latogato van."
```

`bentlevok` property

Hozz létre egy `bentlevok` nevű property-t, amely a látogatók és a biztonsági őrök összesített számát adja vissza.

Készíts egy settert is a `bentlevok` property-hez, amely beállítja a `latogatok` értékét úgy, hogy a `biztonsagi_orok` számát kivonja a megadott értékből, és ezek szerint állítja be a látogatókat (tehát, a `bentlevok` getter property pontosan az imént beállított értéket adja vissza).
Ügyelj rá, hogy legfeljebb a maximális férőhely legyen beállítva látogatók számának.

Teszteléshez használható kód:

```
fesztival = Fesztival("SummerFest")

print(fesztival.nev)  # "SummerFest"
print(fesztival.ferohely)  # 1000
print(fesztival.latogatok)  # 0
print(fesztival.biztonsagi_orok)  # []

fesztival.biztonsagi_felvetel({"nev": "Kovacs Bela", "orzes_vedes": True})
print(fesztival.biztonsagi_orok)  # ["Kovacs Bela"]

print(fesztival.biztonsag_rendben())  # False

fesztival2 = Fesztival("RockFest", 500)
```

```
fesztival2.latogatok = 300
fesztival2.biztonsagi_felvetel({"nev": "Nagy Istvan", "orzes_vedes": True})
fesztival += fesztival2

print(fesztival2)  # "A RockFest fesztivalon jelenleg 300 latogato van."

fesztival2.bentlevok = 310
print(fesztival2.latogatok)  # 309
fesztival2.bentlevok = 1000
print(fesztival2.latogatok)  # 499
```