Programozás I. Gyakorló feladatsor

SZTE Szoftverfejlesztés Tanszék

2024. tavasz

Általános követelmények, tudnivalók

- A feladat elkészítési határideje: **vasárnap 23:59:59**. Ez szigorú határidő, a Bíró előre megadott időben zár, pótlásra nincs lehetőség.
- A feladatokat számítógép előtt kell megoldani, tetszőleges fejlesztői környezetben, tetszőleges operációs rendszer segítségével.
- Az elkészült programot **20** alkalommal lehet benyújtani, a megadott határidőig.
- Csak a leírásban szereplő osztályokat, metódusokat és adattagokat kell megvalósítani, egyéb dolgokért nem jár plusz pont.
- A feladat megoldása során minden megadott előírást pontosan követni kell! Tehát, ha a feladat leírása szerint egy adattag neve a "elsoFoku", akkor az alábbi elnevezések nem megfelelőek: "elsőFokú", "elsofoku", "elso_foku", "elsőFoq". Ugyanez igaz a metódusok, osztályok elnevezésére is!
- A metódusok esetében a visszatérési típus, a név, módosítók és a paraméterek típusai (és azok sorrendje) kerülnek ellenőrzésre, azonban a paraméterek nevei tetszőlegesek lehetnek.
- A Java elnevezési konvenciókat követni kell (getter/setter elnevezés, toString, indentálás, stb). Abban az esetben is, ha ezt a feladat külön nem emeli ki, az ellenőrzés során erre is építünk.
- A nem forduló kódok nem kerülnek kiértékelésre, ezt utólagosan a gyakorlatvezető sem bírálhatja felül. (Hiszen mindenki rendelkezésére áll a saját környezete, ahol fordítani, futtatni tudja a forráskódot, így feltöltés előtt ezt mindenképpen érdemes megnézni!)
- Az adattagok és konstruktorok hiányában garantáltan 0 pontos lesz a kiértékelés, mert ezek minden teszt alapját képezik.
- Ha végtelen ciklus van a programban, akkor ezt a Bíró ki fogja dobni 3 másodperc után (ha többször is meghívásra kerül ilyen metódus, akkor ez többszöri 3 másodperc, összesen akár 2 perc is lehet). Ilyenkor NE kattints még egyszer a *Feltöltés* gombra, mert akkor kifagyhat a Bíró, csak a böngésző újraindításával lehet megoldani a problémát (emellett elveszik 1 feltöltési lehetőség is).
- Kérdés/probléma esetén a gyakorlatvezetők tudnak segítséget nyújtani.
- **A feladat megoldása során a default csomagba dolgozz, majd a kész forrásfájlokat tömörítve, zip formátumban töltsd fel, azonban a zip fájlt tetszőlegesen elnevezheted!**

  - Zip készítése: Windowson és Linuxon is lehet a GUI-ban jobb klikkes módszerrel tömörített állományt létrehozni (Windowsban pl. a 7-Zip nevű ingyenes program használatával).
  - Linux terminálon belül például a "zip feladat.zip *.java" paranccsal is elkészíthető a megfelelő állomány.

- A feladatokban az alábbi dolgok az alapértelmezettek (**kivéve**, ha a feladat szövege mást mond)
  - az osztályok láthatósága publikus
  - az egész érték 32 bites
  - a lebegőpontos számok dupla pontosságúak
  - az olyan metódusok void visszatéréssel rendelkeznek, amelyeknél nincs specifikálva visszatérési típus.
  - a metódusok mindenki számára láthatóak
  - az adattagok csak az adott osztályban legyenek elérhetőek

- A *riport.txt* és a fordítási log fájlok megtekinthetőek 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/IB204L/FELADAT/hXXXXXX/4/riport.txt`
  3. Az url-ből visszatörölve a 4-esig (`riport.txt` törlése) megkaphatók a 4-es próbálkozás adatai.

- Szövegek összehasonlításánál az egyezés a pontos egyezést jelenti, azaz ha kis-nagy betűben térnek el, akkor már nem tekinthetők egyenlőnek (pl. a "piros" != "Piros")

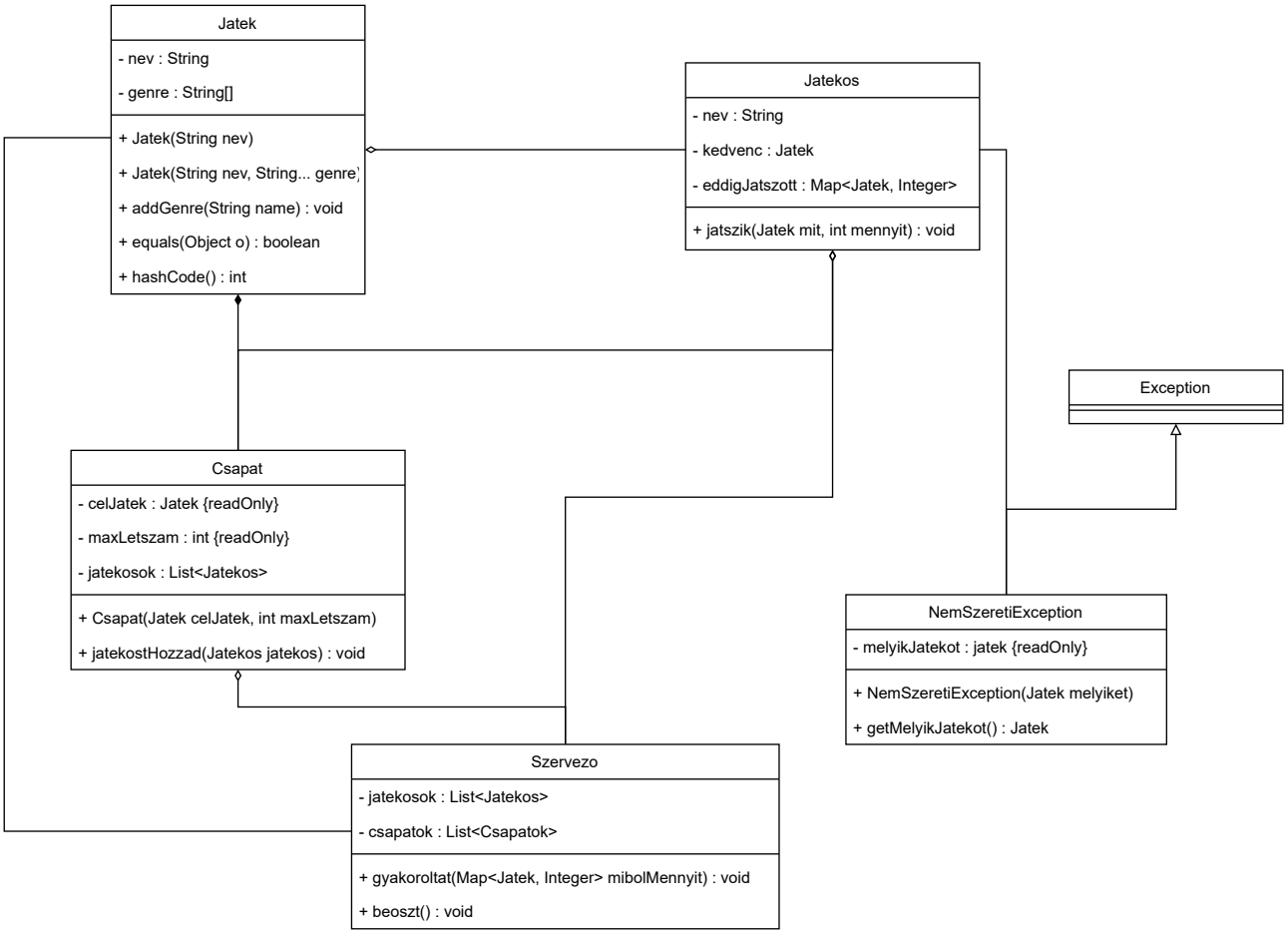
- A leírásokban bemutatott példákban a stringek köré rakott idézőjelek nem részei az elvárt kimenetnek, azok csak a string határait jelölik. Például ha az szerepel, hogy a példa bemenetre az elvárt kimenet az, hogy `"3 alma"`, akkor az elvárt kimenet idézőjelek nélkül a `3 alma`, de a szóköz szükséges!

- Az elvárt kimeneteknek karakterről karakterre olyan formátumúnak kell lennie, ami a feladatban le van írva (szóközöket és sortöréseket is beleértve).

Csapat szervező

Az **??** UML ábrán látható osztály struktúra implementálása a feladatod. A feladat csak az ellenőrzött metódusokat említi, szükséged lehet újak készítésére a teljes megoldáshoz.

1. ábra. UML osztálydiagramok



Jatek(4 pont)

Konstruktorok

Egy játékot 2 módon hozhatunk létre. Ha csak a nevét adjuk meg, akkor egy 3 hosszú üres tömböt állítsunk a genre értékéül, egyébként pedig a tetszőleges számú megadott zsáner közül az első 3-at vegyük be. Ha kevesebb lenne mint 3, a tömb ugyan úgy 3 hosszú legyen, és az első elemétől kezdve legyen feltöltve.

Metódusok

Az `addGenre` metódus hozzáad egy új zsánert az eddig felvettekhez, ha van még rá hely. A tömböt továbbra is az első elemtől kezdve töltsük fel. Ha nem lenne hely az új zsánernek ne történjen semmi.

Fontos! valósítsd meg az equals és a hashCode metódusokat az órán tanult módon. Mindkettő vegye figyelembe mind a genre tömböt, mind a játék nevét. Ezen metódusok nélkül a tesztesetek nem feltétlenül fognak helyesen működni.

NemSzeretiException(3 pont)

Konstruktor

A kivétel mentse el a paraméterül kapott játékot, és állítsa be a következő hibaüzenetet létrejöttekor:

`A␣jatekos␣nem␣szereti␣a␣<jatekNev>␣jatekot.`

Itt a jatekNev a paraméterül kapott játék neve.

Metódus

A `getMelyikJatekot` metódus egyszerűen a letárolt játékkal térjen vissza.

Jatekos(6 pont)

Konstruktor

A játkos nev adattagját állítsa be a paraméterül kapott értékre. A kedvenc játék alapértelmezetten null legyen.

Metódus

A `jatszik` metódus egy játékot és egy időtartamot kap paraméterként, amik azt jelölik, hogy a játékos melyik játékot mennyi ideig próbálja játszani. A játékos csak akkor hajnaldó játszani, ha az eddigi kedvenc játékával legalább egy genre besorolást oszt, egyébb esetben dobj egy `NemSzeretiException`-t, az paraméterül kapott játékkal. Ha a játékosnak nincs még kedvenc játéka, akkor mindenképpen játszik az újjal.

Ha a játékos hajlandó játszani, akkor az `eddigJatszott` leképezésébe vagy fel kell venni a paraméterül kapott értékeket, mint kulcs-érték párt, vagy, ha a játék már szerepelt a leképezésben, frissíteni az eddig játszott órák számát a paraméterül kapottak és az eddigiek összegére.

A játékos minden kör után újraértékeli, hogy melyik a kedvenc játéka, és mindig azt választja, amivel eddig a legtöbbet játszott.

Csapat(3 pont)

Konstruktor

Minden csapat egy bizonyos játékra specializálódok, és csak olyan játékosokat vesznek be, akiknek az a kedvence. A paraméteres konstruktor ezt a játékot, illetve a csapat maximális létszámát várja, és állítsa be.

Metódus

A `jatekostHozzaad` metódus egy csatlakozó játékost vár paraméterként. Ha a játékos kedvenc játéka nem az, ami az adott csapat specializációja, akkor dobjon `IllegalArgumentException` kivételt a következő szöveggel:

`Ezt␣a␣jatekost␣nem␣lehet␣a␣csapathoz␣adni!`

Ha a csapat már elérte a kapacitását, akkor dobjon egy `IndexOutOfBoundsException` kivételt a következő szöveggel:

`Ide␣mar␣nem␣fer␣jatekos!`

Egyébb esetben egyszerűen fűzd a jatekos lista végére a szerencsés felvettet.

Szervezo(8 pont)

A szervező feladata, hogy a játékosoknak csapatot találjon, és ennek érdekében ráveheti a játékosokat, hogy játszanak bizonyos játékokkal, vagy megpróbálhatja csapatba osztani őket.

Metódusok

Az `addJatekos` és az `addCsapat` metódusok a megfelelő lista végére fűzzék a paraméterül kapott értékeket.

A `gyakoroltat` metódus egy Játék-egész szám leképezést vár paraméterül. A szervező ilyenkor az összes, eddig a listájára felvett játékost megpróbálja meggyőzni, hogy próbálják ki a leképezésben szereplő összes játékot, annyi ideig, mint amennyi hozzájuk van rendelve. Tehát minden játékos `jatszik` metódusát a paraméterként kapott legképezés minden kulcs-érték párjával le kell futtatni.

Ha egy játékos egyik játékkal sem hajlandó játszani dobj `SecurityException` kivételt a következő szöveggel:

`Az␣egyik␣jatekos␣tul␣valogatos.`

A `beoszt` metódus nem vár paramétert, és a szervező listáira felvett játékosokat próbálja csapatokba osztani. A beosztás során végig halad minden játékoson, és megpróbálja őket a **csapatok** listáján szereplő csapatokhoz hozzáadni, a `jatekostHozzaad` metódus segítségével. Ha nem sikerült csapatot találni egy játékosnak, mert egyik sem kompatibilis vele (az összes csapat céljátéka eltér a játékos kedvencétől), akkor dobj `Exception` kivételt a következő szöveggel:

`Az␣egyik␣jatekosnak␣nem␣sikerult␣megfelelo␣csapatot␣talalni.`

Ha lenne legalább egy kompatibilis csapat a játékos számára, de az a csapat betelt, akkor szintén dobj `Exception` kivételt, a következő szöveggel:

`Az␣egyik␣jatekosnak␣<mennyit>␣csapatot␣talalt,␣de␣mind␣tele␣van.`

Ahol a mennyit helyére azon csapatok száma kerüljön, amelyekkel a játékos kompatibilis.

Jó munkát!