

# Programozás I.

## Minta Kiugró ZH

A valódi zárthelyi dolgozatban a leírások mellett/helyett előfordulhatnak UML diagramok is. Ezek értelmezése, esetlegesen UML készítése a kész rendszerhez szintén feladat lehet, így ezekre is fel kell készülni!

Az itt található minta feladat nem 100%-ban felel meg a valódi zárthelyinek. Azonban a számonkért tudásanyagot lefedi.

## TamadasNemLehetseges (5 %)

Hozd létre a **TamadasNemLehetsegesException** nevű publikus kivétel osztályt. Az osztályt úgy hozd létre, ha egy metódus ilyen típusú kivételt dobhat, akkor azt mindig jelezni kelljen a függvény fejlécében.

Készíts egy paraméter nélküli konstruktort, amely minden esetben a "Tamadas nem lehetséges!" szöveggel inicializálja őst.

Készíts egy paraméteres konstruktort, amely egy logikai értéket tároló objektumot vár paraméterként. Ha ez az érték igaz, a "Tamadas nem lehetséges!", hamis esetben "Hibas tamadas!", egyéb esetben pedig "Ismeretlen hiba!" szöveggel inicializálja az ősobjektumot.

## Karakter (10 %)

Készíts egy **Karakter** osztályt, amely egy játékbeli karaktert ír le. Az osztály nem példányosítható.

Az osztály rendelkezzen:

- játékos azonosítóval (egész szám), amely megmondja, hogy ez a karakter melyik játékoshoz tartozik,
- névvel (szöveg),
- tapasztalattal (nemnegatív egész szám),
- sebzéssel (egész szám),
- életerővel (egész szám).

Az adattagokat csak az osztályban és a gyerekosztályaiban, valamint csomagon belül lehessen elérni.

Karaktereket úgy szeretnénk létrehozni, hogy az objektum példányosításakor megadjuk a játékos azonosítóját, amelyet ezután nem lehet már megváltoztatni, a karakter nevét, tapasztalati pontjait, sebzését, valamint életerejét. Ügyelj rá, hogy a tapasztalat garantáltan nemnegatív értéket vegyen fel. Ha mégis negatív értékkel próbálnánk meg inicializálni egy karaktert, akkor dobj *IllegalArgumentException* típusú kivételt!

A karakterek szöveggé alakíthatóak. Szöveggé alakításkor a "<név> <szám> xp" legyen a visszaadott érték, ahol a <név> a karakter neve, a <szám> pedig a karakter tapasztalata.

Az osztálynak legyen egy **tamad** metódusa, amely egy másik karaktert vár paraméterként, és logikai értékkel tér vissza. A metódus *TamadasNemLehetsegesException* típusú kivételt dobhat. A metódus ne legyen megvalósítva.

Készítsd el a **jatekosEllenorzes** metódust, amely paraméterben egy játékos azonosítót kap és leellenőrzi, hogy megegyezik-e az adott karakterben található játékos azonosítóval (tehát logikai értékkel tér vissza a függvény)!

Készíts egy **sebzestElszenved** metódust, amely paraméterben egy számot vár, és ennyivel csökkenti a karakter életerejét! Amennyiben így az életereje 0-nál kisebb értékre csökkenne, állítsd 0-ra!

Készítsd el a **meghalt** metódust, amely logikai értékkel tér vissza és nem vár paramétert! Térjen vissza igazgal, ha a karakter életereje 0!

Készíts egy **getSzint** nevű metódust, amely nem vár paramétert, és egy egész számmal tér vissza: a karakter szintjével! A karakter szintjét úgy számítjuk ki, hogy vesszük a tapasztalatának gyökét, elosztjuk 10-zel. Ezután hozzáadunk az eredményhez 1-et, majd vesszük az eredménynek az egész részét!

## Íjász (5 %)

Készítsd el az **Ijasz** osztályt! Az íjászok példányosítható karakterek.

Egy íjász tudja magáról, hogy hány darab nyila van, ami mindig egy nemnegatív érték. Ezt az információt azonban nem lehet kívülről elérni, csak az osztályon belülről.

Készíts egy konstruktort, amely a paraméterek alapján inicializálja az adattagokat. A konstruktor az ős tulajdonságain kívül várja a nyilak számát is!

Oldd meg, hogy a nyilak számát ne legyen kötelező megadni. Ebben az esetben 5 nyila legyen a létrejövő íjásznak!

Készítsd el a **tobbNyil** metódust, ami egy másik íjászt vár paraméterül, és igazgal tér vissza, ha az adott íjásznak több nyila van, mint a paraméterben kapott íjásznak!

Valósítsd meg a **tamad** metódust! A metódusban az íjász rá fog lőni a paraméterben kapott karakterre.

- Ha az íjásznak nincs több nyila, akkor dobj egy *TamadasNemLehetsegesException* típusú kivételt!
- Amennyiben van még nyila, akkor történjen meg a támadás, csökkentsd a nyilak számát, és a paraméterben kapott karaktert sebez meg!
- Ezután az íjásznak kapjon sebzés\*13 tapasztalati pontot!
- Végül ellenőrizd, hogy a paraméterben kapott karakter meghalt-e. Ha igen, akkor írjunk ki egy üzenetet a standard kimenetre, amivel jelezzük, hogy az íjász legyőzte ellenfelét.

A függvény térjen vissza igazgal, ha a paraméterben kapott karakter meghalt, hamissal, ha nem.

## Kardos (10%)

A Kardos is egy speciális karakter. Az osztály az őstől kapott tulajdonságokon kívül rendelkezzen **kardok száma** tulajdonsággal, ami megmondja, hogy hány kardja van. Ez 0, 1, vagy 2 lehet. Ez az információ sem publikus használatra van szánva.

A kardok beállítására legyen lehetőség, amely hibás érték esetén a standard hibakimenetre kiírja a "Hiba!" szöveget.

Készíts egy konstruktort, amely a paraméterek alapján inicializálja az adattagokat. A konstruktor az ős tulajdonságain kívül várja a kardok számát is!

A Kardos létrehozható egy pontosvesszővel elválasztott Stringből is. Készíts egy ilyen konstruktort is, amely a paraméterben kapott szöveget feldolgozza, konvertálja, majd meghívja a már létező másik konstruktort!

Készítsd el a **kardotHozzaad** metódust! Ez nem vár paramétert, és egy logikai értékkel tér vissza. A metódus növelje meg a kardok számát abban az esetben, ha az még nem érte el a 2-t! Ellenkező esetben ne csináljon semmit! A metódus térjen vissza igazgal, ha sikerült a művelet, ha pedig már eredetileg is 2 kardja volt, akkor hamissal térjen vissza!

Készítsd el a **tamad** metódust. A metódusban a kardos megtámadja a paraméterben kapott karaktert.

- Ha a kardosnak nincs egy kardja sem, akkor dobj *TamadasNemLehetsegesException* típusú kivételt!
- Egyéb esetben a paraméterben kapott karakternek hívjuk meg a *sebzestElszenved* metódusát. A sebzés mértéke megegyezik a kardos sebzésével abban az esetben, ha egy kardja van. Amennyiben viszont 2 kardja van, akkor másfélszer akkora értéket adjunk át a metódusnak (lefele kerekítve)!
- Növeld a kardos tapasztalatát az okozott sebzés 10-szeresével!
- Térjen vissza igazgal a függvény, ha a paraméterben kapott karakter meghalt, egyéb esetben hamissal!

## Jatek (70%)

Készítsd el a Jatek osztályt. Egy játékról tároljuk el a nevét. Ez publikusan elérhető információ. Tároljuk el még a játékban résztvevő csapatokat (egy csapatról csak a nevét tudjuk, ami szöveg), és az egyes csapatokban lévő karaktereket. A karakterek eltárolását mindig azonosítójuk alapján rendezetten tegyük meg, növekvő sorrendben. Természetesen mind a csapatok, mind a csapatban lévő karakterek dinamikusan módosulhatnak a program futása szerint, ezt vedd figyelembe!

Készíts egy konstruktort, ami csak a játék nevét várja paraméterben, ezt inicializálja!

Készítsd el a **karaktertHozzaad** metódust, amely paraméterben egy csapatot, valamint egy Karakter típusú objektumot vár, és nem tér vissza semmivel! A metódus adja hozzá a kapott karaktert az adott csapathoz! Ha a csapat nem létezik, hozd létre!

Készítsd el a **csapatTorles** metódust, amely a paraméterben kapott nevű csapatot kitörli (amennyiben létezik). Amennyiben nem létezik, ne csináljon semmit.

Készítsd el a **nevsor** metódust, ami minden csapat minden karakterének a nevét elmenti a "nevsor-<csapat>.txt" fájlba, ahol a <csapat> az adott csapat neve! A metódus ne térjen vissza semmivel és minden hibakezelést valósítson meg!

Készítsd el az **osszeszamol** metódust, amely nem vár paramétert, és szöveggel tér vissza! A metódus határozza meg, hogy a játékban íjászból vagy kardosból van több. Ennek megfelelően "íjász" vagy "kardos" szöveggel térjen vissza!

Készítsd el a **legjobbKarakter** nevű metódust, ami nem vár paramétert, és egy Karakter típusú objektummal tér vissza! A metódus határozza meg a legmagasabb szintű karaktert, és térjen vissza ezzel a karakterrel! Amennyiben nincs egy karakter sem, akiből választhat, úgy null értékkel térj vissza!

Készítsd el a **csapattagokTapasztalata** nevű metódust. Egy darab szöveges paramétert vár a metódus, és nemnegatív egészeket tároló tömbbel tér vissza. A metódus az adott csapatba tartozó karakterek tapasztalat értékeit adja vissza egy tömbben. Amennyiben a csapat nem létezik, vagy üres, akkor a metódus egy üres tömböt adjon vissza!