3. feladatsor

Határidő Nincs megadva határidő **Pont** 0 **Kérdések** 1 **Időkorlát** Nincs
Engedélyezett próbálkozások Korlátlan

Kvíz kitöltése újra

Próbálkozások naplója

	Próbálkozás	Idő	Eredmény
LEGUTOLSÓ	1. próbálkozás	387 perc	0 az összesen elérhető 0 pontból *

* Néhány kérdés még nem lett értékelve

Beadva ekkor: feb 22, 20:50

incs megválaszolva **érdés** **Még nincs értékelve / 0 pont**

Programozási nyelvek (BSc, 18) Java 3. feladatsor

1. feladat

Szervezze az előző órai `Point` osztályt és az őt bemutató főprogramot a `point2d` csomagba. A `Point` osztály ne látszódjon ki a csomagból. A `Point` osztály és a főprogram kerüljenek külön fordítási egységekbe.

2. feladat

Módosítsa az előző megoldást úgy, hogy a `Point` osztályt bemutató főprogramot átszervezi egy másik, `pointm` csomagba.

3. feladat

a

Készítsen `Circle` osztályt, amellyel egy síkbeli kört reprezentálunk. Egy körnek van középpontja, `double` típusú `x`, `y` adattagja, amelyeket inicializáljunk 0-ra, illetve sugara (`radius`) amelyet inicializáljunk 1-re. Írjon `getArea()` metódust, amely visszatér a kör területével. Példányosítson egy kör objektumot, írjuk ki a területét, maj dállítsuk be a középpontját (5, 2)-re, sugarát 10-re, majd írja ki a képernyőre a kör területét.

b

Módosítsuk az **a** megoldást úgy, hogy az adattagokhoz csak megfelelően megírt getter és setter metódus férhessen hozzá. A kör sugara nem lehet 0 vagy negatív szám, ilyenkor a setter metódus dobjon `IllegalArgumentException` kivételt.

c

Módosítsuk a **b** megoldást úgy, hogy az adattagok beállítását a `Circle` osztály konstruktárval végezzük.

4. feladat

a

A korábban megírt `Point` osztályt és a 3. feladat **c** megoldását szervezzük csomagokba.

A `Circle` osztály legyen a `circle` csomagban, az őt bemutató főprogram `CircleMain` néven szintén a `circle` csomagban, valamint a `Point` osztály a `circle.utils` csomagba legyen szervezve. A `circle.utils.Point` osztály `double` típusú `x`, `y` adattagokat és getter/setter metódusokat tartalmaz. A `circle.Circle` osztálynak legyen egy `getCenter()` metódusa, amely visszatér a kör középpontjával `circle.utils.Point` típusú pontban tárolva.

b

Módosítsa az **a** megoldást úgy, hogy a `Point` osztály setter metódusok helyett konstruktorral végzi az adattagok beállítását.

5. feladat

Készítsen `stringutils` néven csomagot. A `stringutils.IterLetter` osztály konstruáláskor fogadjon egy `String` referenciát (kezeljük azt az esetet, ha ez `null`). Az osztálynak legyen egy `printNext()` metódusa, amellyel új sorban a képernyőre írjuk a sztring következő karakterét.

Ha a sztring összes karakterét kiírtuk a képernyőre, akkor a metódus többé ne írjon ki semmit. Az osztálynak legyen egy `restart()` metódusa, amely hatására a következő `printNext()` hívás a sztring elejét kezdi el kiírni. Az osztálynak legyen egy `hasNext()` metódusa, amely `true` értékkel tér vissza, ha van még kiírható elem.

Készítsen `Main` néven főprogramot, amely legyen névtelen csomagban. A főprogram példányosít egy `stringutils.IterLetter` osztályt, majd bemutatja annak használatát.

1. gyakorló feladat

A `Kangaroo` osztály egy kengurut reprezentál. Az osztálynak két adattagja van, az egyik egy szöveges típusú, a kenguru nevének, a másik egész

típusú és az életkorának az eltárolására szolgál.

Az osztálynak két konstruktora van. Az első egy szöveges típusú nevet és egy egész típusú életkort kap paraméterként és beállítja a megfelelő adattagokat. A második konstruktor egy egész típusú értéket kap és kiírja a kenguru lábainak számát. Az osztály rendelkezik egy `display()` metódussal is, egy szöveges típusú országnevet kap paraméterül, és kiírja a kenguru nevét, lakóhelyét (az országot), majd eggyel megnöveli az életkorát és az új életkort is kiírja.

2. gyakorló feladat

Bővítse a `stringutils` csomagot a `stringutils.IterWord` osztállyal; az osztály konstruáláskor fogadjon egy sztringet. Az osztály `printNext()` metódusa új sorban a képernyőre írja a sztring következő szavát. Az osztálynak szintén legyen `restart()` és `hasNext()` metódusa.

Készítsen `Main` néven főprogramot, amely bemutatja az osztály használatát.

3. gyakorló feladat

Készítsen `Book` osztályt, amellyel egy könyvtári könyvet reprezentálunk. Egy `Book`-nak legyen címe (`String`), szerzője (`String`), kiadás éve (`int`), oldalszáma (`int`) és egy logikai jellemzője, hogy kikölcsönözhető-e (`boolean`).

Írjon olyan konstruktort, amely minden jellemzőjét fogadja a könyvnek; illetve olyan konstruktort is, amely paraméterként egy másik `Book` referenciát vár (figyeljen arra az esetre, ha `null` referenciát kap).

Készítsen `toString()` metódust, amellyel sztringgé alakítja az objektum belső állapotát, valamint egy `equals()` metódust is, amellyel eldönti, hogy a paraméterként kapott `Book` referencia ugyanazt a könyvet reprezentálja-e (két `Book`-ot akkor tekintünk egyenlőnek, ha ugyanaz a címe, szerzője, kiadás éve és oldalszáma; a kikölcsönözhetőség tehát nem lényeges).

Készítsen főprogramot, amellyel bemutatja az osztály használatát, valamint szervezze az osztályt és a főprogramot a `library` csomagba.

4. gyakorló feladat

Készítsen egy `game.utils.Vehicle` osztályt, amellyel egy MultiPlayer-es játék járművét reprezentáljuk. Egy járműnek van modelid-je (`int`), rendszáma (`String`), és két színállapota (`color1`, `color2` `int` típusú adatok). A rendszámhoz készítsen setter és getter metódusokat.

Készítsen `game.Player` osztályt, amellyel egy MultiPlayer-es játék felhasználóját (játékos) reprezentálunk. Egy játékosnak van neve (`String`), IP-címe (`String`), egészségi állapota (`int`) és lehet járműve (`game.utils.Vehicle`) (ha nincsen, akkor tároljunk `null` értéket).

A játékos osztályhoz készítsen `toString()` metódust, amellyel sztringgé alakítjuk egy játékos legfontosabb információit: nevét, IP-címét, egészségi állapotát, illetve járművének rendszámát (ha van).

Készítsen `game.Main` főprogramot, amelyben példányosít legalább 3 járművet, legalább 2 játékost; az egyik játékoshoz tartozzon jármű; írja ki a képernyőre a játékosok adatait.

5. gyakorló feladat

Alakítsa át a korábban megírt `Segment` osztályt, vagy készítse el a `Segment` osztályt az alábbiaknak megfelelően. A `Segment` objektumok egy szakasz két végpontjának koordinátáját tárolják négy lebegőpontos típusú adattagban: `x1`, `y1`, `x2` és `y2`. Csak valódi szakaszokkal szeretnénk dolgozni, ehhez az kell, hogy az `(x1,y1)` és `(x2,y2)` pontok ne essenek egybe. A típusinvariáns biztosításához használjon információelrejtést, valamint ellenőrzéseket végző konstruktort és *setter* műveleteket!

Készítsen `midpoint()` metódust a `Segment` osztályba, mely megadja egy szakasz felezőpontját egy `Point` objektum formájában.