Programozási nyelvek (BSc, 18) Java 11. gyakorlat

1. feladat

Módosítsa a 9. gyakorlat 2. feladatát a következők szerint. `Book` objektumot a következőkben ne lehessen konstruálni, csak a leszármazott osztályait, továbbá a `Book` osztály garantája azt, hogy a leszármazott osztályok interfésze legalább a következő: konstruktor, `getShortName()`, `toString()`, `getPrice()`, `createReference()`.

A `getPrice()` metódus tartozzon a `Book` osztályba, de ne tartozzon hozzá implementáció. Az osztály legyen absztrakt; a `Book` osztály implementálja a következő metódusokat:

- `getShortName()`
- `toString()`

Az `Book` osztály nem implementálja a következő metódusokat (ezek implementálása a leszármazottakban történjen):

- `getPrice()`
- `createReference(String, int, int)`

2. feladat

Ez a fiktív Java program több osztályból áll, amely osztályoknak egyebek mellett két tulajdonságot kell biztosítani. Egy objektum állapota visszafordítható (reversable), ha van `reverse()` metódusa, amely az objektum belső állapotát a legutolsó `set` hívás előtti állapotra állítja. Például, ha egy pontot reprezentáló `ReversablePoint` -nak x és y adattagja van `(2, 3)` értékkel, akkor egy `setX(10)` settert követő `reverse()` hívás után x és y értéke újra `(2, 3)`.

A nyomtatható (printable) tulajdonság azt jelenti, hogy egy osztálynak van `print()` metódusa, amely a képernyőre írja az osztály belső állapotát. Természetesen egy osztály rendelkezhet mindkét tulajdonsággal is (`PrintableAndReverseablePoint`).

a

Készítsen `Reversable` néven `interface` -t, amely egy `reverse()` nevű, paraméter nélküli, visszatérési érték néküli metódust tartalmaz. Írjon `ReversablePoint` néven osztályt, amely x és y egész jellegű pontokat ábrázol, amely megvalósítja a `Reversable interface` -t.

Írjon `Printable` néven `interface` -t, amely egy `print()` nevű, paraméter nélküli, visszatérési érték néküli metódust tartalmaz. Írjon `PrintablePoint` néven osztályt, amely x és y egész jellegű pontokat ábrázol, amely megvalósítja a `Printable interface` -t (ezt a lépést át lehet ugrani, és ez a kód rögtön kiemelhető egy `Point` ősosztályba, ld. (b) feladat). Írjon `Book` osztályt, amely szintén megvalósítja a `Printable interface` -t. Egy könyvnek szerzője, címe és konstruktora van. Írjon `foo()` statikus metódust a főprogramba, amely nyomtatható objektumokat fogad paraméterként, és meghívja a `print()` metódusát.

Írjon `PrintableAndReverseablePoint` néven osztályt, amely megvalósítja a `Printable` és `Reversable interface` -eket.

Megvalósítható-e egy ilyen osztályszervezés absztrakt osztályokkal?

b

Módosítsa az (a) megoldást úgy, hogy a `ReversablePoint` , `PrintablePoint` és `PrintableAndReverseablePoint` osztályokból kiemeli a közös kódokat egy `Point` ősosztályba (x, y adattag; konstruktor; getter; setter). A leszármazott osztályok override-olják az ősosztály setterét, egyéb teendők mellett hívják meg az ősosztály setterét.

c

Szervezzen minden osztályt és `interface` -t külön Java fordítási egységbe.

1. gyakorló feladat

Hozzunk létre egy absztrakt `Prism` osztályt, amelynek a segítségével hasábokat tudunk ábrázolni! Tároljuk el benne a hasáb magasságát (`height`), valamint legyen egy olyan absztrakt (vagyis a leszármazottakban megvalósítandó) metódus, amely az alapterületét számolja ki (`baseArea()`). Ennek felhasználásával aztán készítsünk egy másik metódust (`volume()`), amely a hasáb térfogatát számítja ki a magasság és az alapterület segítségével. Tegyük a `Prism` osztályt a `polyhedra` csomagba! A hasábokból származtassuk a hengereket ábrázoló `Cylinder` osztályt, illetve a kockákat ábrázoló `Cube` osztályt!

Az absztrakt metódus implementációja mellett definiáljuk felül azok `toString()` metódusait, hogy a típusnak megfelelő szöveges reprezentációval térjenek vissza.

`Cylinder` esetén:
`Cylinder : (h=10 , r=5)`

`Cube` esetén:
`Cube : (h=4)`

Ezek az osztályok is kerüljenek a `polyhedra` csomagba!

2. gyakorló feladat

Írjon `Shape` néven `interface` -t, amely tetszőleges alakzat kerületét és területét kiszámító metódusok szignatúráit tartalmazza: `getPerimeter()` , `getArea()` .

Írjon `Square` , `Rectangle` , `Circle` néven négyzetet, téglalapot és kört reprezentáló osztályokat, amelyek megvalósítják a `Shape interface` -t.