OBJEKTUMORIENTÁLT PROGRAMOZÁS 2008.
5. GYAKORLAT

CÉL:

- Származtatás (örökítés). Metódusok felülírása.
- Heterogén tárolók és polimorfizmus.

Folytatjuk az előző órákon készített bankos projektet!

1. Feladat

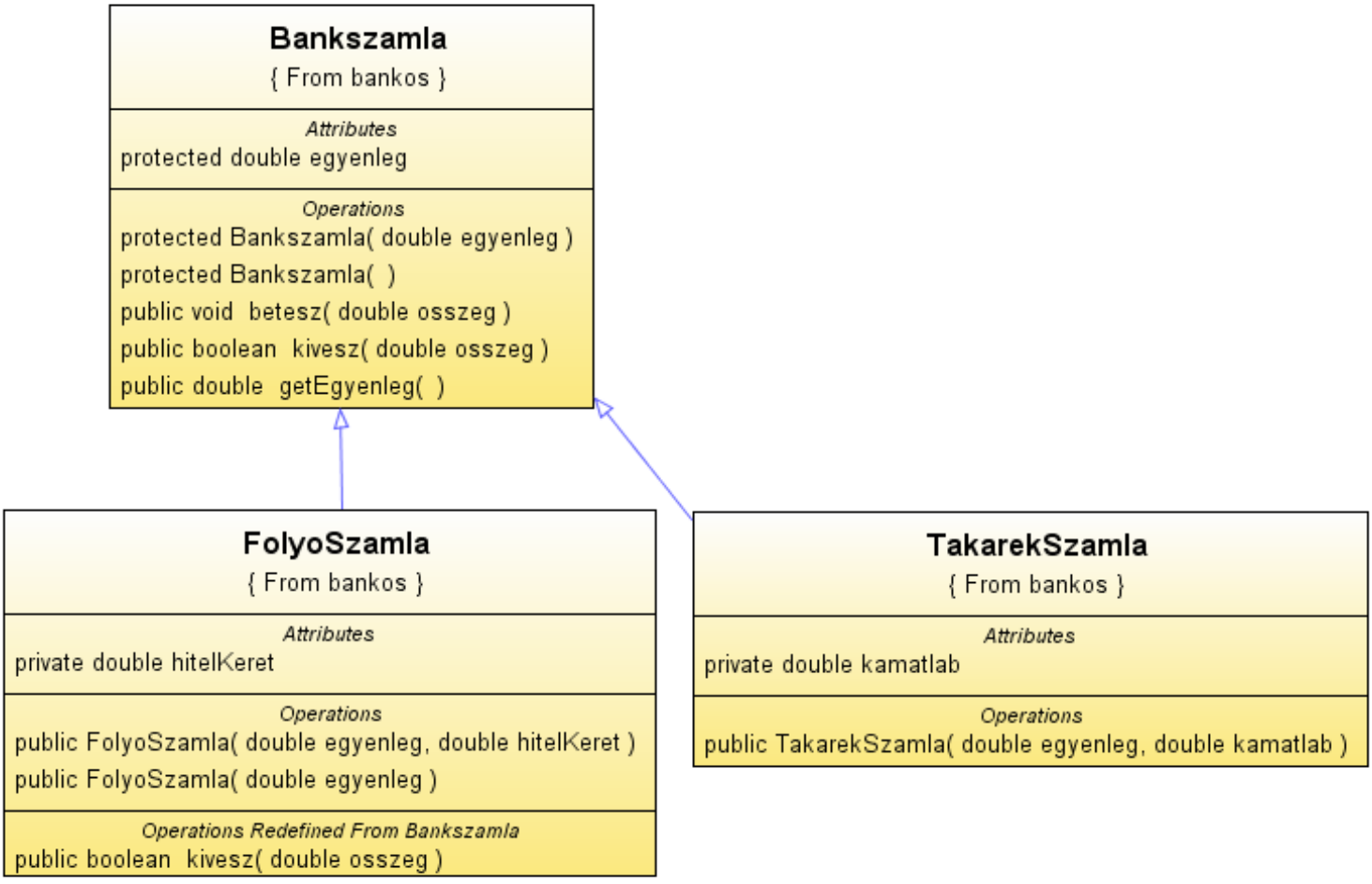
1. lépés

Előkészítjük a **`Bankszamla`** osztályt származtatásra:

- változtassa meg az `egyenleg` láthatósági modosítóját: **`private--> protected`**
- változtassa meg a **konstruktorok** láthatósági módosítóját: **`public--> protected`**

2. lépés

Készítsen két származtatott osztályt a `Bankszamla` osztályból, egy takarék-, illetve egy folyószámlát. Használja a **`TakarekSzamla`** és **`FolyoSzamla`** megnevezéseket.



OBJEKTUMORIENTÁLT PROGRAMOZÁS 2008.
5. GYAKORLAT

- Származtatás Java:
  public class TakarekSzamla extends Bankszamla{...}

- `A TakarekSzamla` osztály konstruktorának két paramétere van. Az első `egyenleg` nevű paramétert az ősosztály konstruktorának hívásához használjuk, a második `kamatlab` paramétert pedig a `kamatlab` attribútum kezdőértékének.

- Ősosztály konstruktorának hívása:
  `super( paraméter1, paraméter2, ..., paramétern)`

- A `FolyoSzamla` osztálynak két konstruktora van. Az egyparaméteres konstruktor esetében a `hitelKeret` attribútum értéke `0`.

- A `kivesz` metódust azért kell felülírni, mert hitelkeret esetén megengedett, hogy az `egyenleg+hitelKeret` legyen a felső határa a kivett összegnek.

3. lépés

Tesztelje le az alkalmazást:

- hozzon létre egy Bank példányt
- adjon a bankhoz egy **Árva Panna** nevű ügyfelet
- adjon hozzá **Árva Panna** ügyfélhez egy folyószámlát 4000 RON kezdőösszeggel és 1000 RON hitelkerettel
- Árva Panna vegyen ki 4500 RON-t, majd írassuk ki a számláján levő összeget

- adjon a bankhoz egy **Babszem Jankó** nevű ügyfelet
- adjon hozzá **Babszem Jankó** ügyfélhez egy folyószámlát 500 RON kezdőösszeggel
- adjon a bankhoz egy **Babszem Juliska** nevű ügyfelet
- adja hozzá **Babszem Juliskához**, férje, **Babszem Jankó** számláját (megosztva használják!!!)
- **Babszem Jankó** tegyen be 1000 RON-t a számlájára, majd írassuk ki a számláján levő egyenleget
- **Babszem Juliska** vegyen ki 200 RON-t, majd írassuk ki a számláján levő egyenleget

KÉRDÉSEK:

1. **Miért lehetséges az ügyfél számlájának megosztása?**
2. **Lehetséges-e `Bankszamla` típusú példány létrehozása a `bankos` csomagon kívül?**

2. Feladat

OBJEKTUMORIENTÁLT PROGRAMOZÁS 2008.
5. GYAKORLAT

Egy ügyfélnek több, esetleg különböző típusú számlája is lehet. Az aktuális alkalmazás ezt nem teszi lehetővé. Módosítani fogjuk az **Ugyfel** osztályt úgy, hogy lehetővé tegyük több számla felvételét. Mivel egy ügyfélnek több típusú számlája lehet az **egy-sok** kapcsolat megvalósítását úgy kell végeznünk, hogy ezt lehetővé tegyük.

1. lépés

Módosítsa az **Ugyfel** osztályt:

- új attribútumok
  - konstans: int MAXSZAMLASZAM = 10, ez lesz a szamlák tömbjének kapacitása
  - `szamlakSzama:` privát, egész típusú attribútum, amelyben az adott ügyfél számláinak számát tároljuk:
  - `szamlak:` privát, `Bankszamla` tömb típusú attribútum, amelyben a számlákat tároljuk (Miért Bankszamla típust használunk?)
- módosítsa a konstruktort
  - inicializálja a `szamlakSzama` attribútumot
  - végezze el a helyfoglalást a tömbnek

- módosítsa a `setSzamla` metódus nevét --->**`ujSzamla`**

- az `ujSzamla( Bankszamla szamla )` metódus, a paraméterként kapott számla objektumot helyezze be a szamlak tömbbe

- vezessen be egy új metódust a számlák számának lekérdezésére: getSzamlakSzama()

- módosítsa a `Bankszamla getSzamla()` metódust --> `Bankszamla getSzamla( int index)`

2. lépés

Tesztelje az alkalmazást!!! Írja újra a main metódust: legyen a banknak legalább három ügyfele, minden ügyfélnek legyen legalább egy takarék és egy folyószámlája. Adjon minden esetben a számláknak kezdőegyenleget, folyószámla esetén pedig használjon hitelkeretet. Írassa ki a bank ügyfeleit a következő formátumban:

```
Ügyfél neve
     Bankszámla típusa   Egyenleg
     Bankszámla típusa   Egyenleg
     ...
Ügyfél neve
     Bankszámla típusa   Egyenleg
     Bankszámla típusa   Egyenleg
```