Java – Összetett OOP feladat – Banki szolgáltatások

1.Feladat Készíts el egy új projektet **BankiSzolgaltatasok** néven.

a.) Ügyelj rá, hogy a feladat megoldása során ékezetmentes azonosítókat használj.

b.) A projekt megoldása során egy git repoban dolgozz. A projekt távoli eléréséhez githubon hozz létre egy repot „SajatNev_BankiSzolgaltatsok" formában. pl.: ha a neved Gipsz Jakab Zoltán, akkor a repo neve GipszJakabZoltan_BankiSzolgaltatasok legyen.

c.) Az elkészítendő projekt UML diagramja megtalálható a csatolt fájlok között.

2.Feladat Készíts **Tulajdonos** osztályt

a.) Kívülről írható/olvasható formában tárolja el a tulajdonos nevét

b.) Biztonsági okokból ne lehessen belőle származtatni

3.Feladat Készíts **BankiSzolgáltatás** osztályt

a.) A konstruktorban lehessen megadni a tulajdonost, ez a későbbiekben csak olvasható legyen

b.) Ebből az osztályból ne lehessen közvetlenül példányosítani

4.Feladat Készíts **Számla** osztályt

a.) Legyen a BankiSzolgáltatásosztály leszármazottja

b.) Konstruktorában lehessen megadni a tulajdonost

c.) Kívülről csak olvasható formában tárolja el az aktuális egyenleget

d.) Egy Befizet(összeg) metódussal lehessen növelni az egyenleget

e.) Legyen egy hasonló paraméterű, de nem implementált Kivesz(összeg) metódusa is, aminek a visszatérési értéke egy logikai érték

5.Feladat Készíts **HitelSzámla** osztály

a.) Legyen a Számlaosztály leszármazottja

b.) A konstruktorban lehessen megadni a tulajdonos mellett a hitelkeret összegét, a későbbiekben ez csak olvasható legyen

c.) Valósítsa meg úgy a Kivesz(összeg) metódust, hogy csak a hitelkeret mértékéig engedjen negatív számla egyenleget. Ellenkező esetben ne csökkentse az egyenleget és hamis visszatérési értékkel jelezze, hogy nem sikerült a kivétel

6.Feladat Készíts **MegtakarításiSzámla** osztályt

a.) Legyen a Számlaosztály leszármazottja

b.) Kívülről írható/olvasható formában tárolja el a kamat mértékét

c.) Az osztály egy statikus mezőjében tárolja el az alapértelmezett kamatot. Egy új megtakarítási számla létrehozásakor ez legyen a kamat kezdőértéke

d.) A Kivesz(összeg) metódus ne engedje 0 alá csökkenni az egyenleget, visszatérési értéke jelezze, hogy sikerült-e a kivét

e.) Legyen egy Kamatjóváírás() metódusa, ami jóváírja az esedékes kamatot

7.Feladat Készíts **Kártya** osztályt

a.) Legyen a BankiSzolgáltatásosztály leszármazottja

b.) A konstruktorban lehessen megadni a tulajdonos mellett a hozzá tartozó mögöttes számlát, illetve a kártya számát

c.) A kártyaszám legyen kívülről olvasható, a mögöttes számla nem módosítható

d.) Készítsen egy Vásárlás(összeg)metódust, ami a paraméterként megadott összeggel megpróbálja csökkenteni a mögöttes számla egyenlegét, és visszatérési értéke legyen ennek sikeressége

8.Feladat Egészítsd ki a **Számla** osztályt

a.) Egészítse ki a Számlaosztály egy ÚjKártya(kártyaszám)metódussal, amely a leendő kártyaszámot várja paraméterként

b.) A metódus hozzon létre egy új kártyát (az aktuális számlát és annak tulajdonosát adva meg a kártya adataiként) és legyen ez a metódus visszatérési értéke

9.Feladat Készíts **Bank** osztályt

a.) Tároljon el tetszőleges számú számlát, ezek maximális számát a Bankkonstruktorában lehessen megadni

b.) Legyen egy Számlanyitás(tulajdonos, hitelkeret)metódusa, amelynek paraméterei egy Tulajdonos objektum és egy hitelkeret összeg. A hitelkeret összegének megfelelően hozzon létre hitel vagy megtakarítási számlát, ezt tárolja el, és ez legyen a metódus visszatérési értéke is

c.) Legyen egy Összegyenleg(Tulajdonos)metódusa, amely visszaadja a paraméterként átadott tulajdonos számláinak összegyenlegét

d.) Legyen egy LegnagyobbEgyenlegűSzámla(Tulajdonos)metódusa, amely visszaadja a megadott tulajdonos legnagyobb egyenlegű számláját

e.) Legyen egy Összhitelkeret()metódusa, amely visszaadja a bank által az összes ügyfélnek adott hitelkeretek összegét

10.Feladat A fenti osztályok implementálását követően hozzon létre példa Tulajdonos és Bank objektumokat, majd próbáld ki a fenti funkciók működését