Zadatak 3

Implementirati interfejs Banka koja sadrzi:

IRacun NoviRacun(IOsoba osoba, Decimal dozvoljeniMinus);

Implementacija treba da sadrži memorijsku strukturu za skladištenje računa korisnika:

Dictionary<IOsoba, List<Racun>> racuni

I treba da sadrži identifikator banke koji je tipa string.

Konstruktor treba da primi ID banke.

- ID banke je trocifren broj koji ne sadrži nule

Realizovati funkciju NoviRacun koja prima objekat Vlasnik I dozvoljeni minus I proverava:

- Vlasnik ne sme biti null
- Dozvoljeni minus mora biti veći od 0

Funkcija treba da proveri da li postoji osoba u računima, ukoliko ne postoji dodati je I njen račun, ukoliko postoji pridružiti joj novi račun.

Napisati test slučajeve koji pokrivaju slučajeve:

- Konstruktor prima dobre parametre
- Konstruktor prima granične paramete
- Konstruktor prima loše parametre
- Funkcija NoviRacun prima dobre parametre
- Funkcija NoviRacun prima ganične parametre
- Funkcija NoviRacun prima loše parametre

Koristiti Mock za interfejs Osoba prilikom testiranja Računa

Zadatak 4

Implementirati interfejs Transakcija

VrstaTransakcije TipTransakcije (enumeracija – prenos, uplata i isplata)
IRacun RacunUplatioca
IRacun RacunPrimaoca
Decimal Iznos

Implementirati 2 konstruktora.

- Prvi konstruktor treba da primi prilikom inicijalizacije vrstu transakcije, racun uplatioca, racun primaoca i iznos.
  - Prilikom inicijalizacije objekta klase Transakcija, racun uplatioca i racun primaoca ne smeju biti null, vrsta transakcije mora biti prenos i iznos mora biti veći od 0.
- Drugi konstruktor treba da primi prilikom inicijalizacije vrstu transakcije, racun i iznos.
  - Prilikom inicijalizacije objekta klase Transakcija, racun ne sme biti null, vrsta transakcije mora biti uplata ili isplata i iznos mora biti veći od 0.

Napisati test slučajeve koji pokrivaju slučajeve:

- Konstruktor prima dobre parametre
- Konstruktor prima granične paramete
- Konstruktor prima loše parametre