```java
import lt.itakademija.exam.*;

import java.util.ArrayList;
import java.util.List;

public class BankImpl implements Bank {

    private final CurrencyConverter currencyConverter; //deklaruojam valiutos keitykla
    private final List<Customer> customers; //deklaruojam lista kad butu kur deti vartotojus, sukuriamus pagal metoda 'createCustomer'

    //trys generatoriai eiliskumui-unikalumui:
    private SequenceGenerator customerSequence = new SequenceGenerator(); //sukuriam generatoriu kad vartotojas turetu eiliskuma
    private SequenceGenerator accountSequence = new SequenceGenerator(); //sukuriam generatoriu saskaitai kita valiuta
    private SequenceGenerator operationSequence = new SequenceGenerator(); //sukuriam generatoriu operaciju eiliskumui


    public BankImpl(CurrencyConverter currencyConverter) { //'Generate --> Construktor'
    this.currencyConverter = currencyConverter;
    this.customers = new ArrayList<>(); //rankom (buvo this.customers = customers)
    }

    @Override // visi override generuojami, o paskui koreguojami rankomis
    public Customer createCustomer(PersonCode personCode, PersonName personName) {
        if (personCode == null || personName == null) { //cia tik patikrinimas ar vartotojas turi duomenis (koda ir varda)
            throw new NullPointerException();
        }

        for (Customer customer : customers) { //cia tik patikrinimas ar vartotojas nesikartoja (ar toks jau yra ar nera)
```

```
29              if (customer.getPersonCode().equals(personCode)) {  //tikrinam pagal koda (nes jis unikalus)
30                  throw new CustomerCreateException("Customer already exist");
31              }
32          }
33
34          Customer customer = new Customer (customerSequence.getNext(), personCode, personName); //sukuriam
    vartotoja + eiliskumas
35          customers.add(customer); //pridedam vartotoja i lista
36          return customer;
37      }
38
39      @Override
40      public Account createAccount(Customer customer, Currency currency) {
41          if (customer == null || currency == null) { //patikrinimas ar saskaita turi duomenis (vartotoja ir
    valiuta)
42              throw  new NullPointerException();
43          }
44
45          if (!customers.contains(customer) ) { //patikrinimas ar vartotojas jau turi saskaita kokia nors kita
     valiuta (pvz. EUR)
46              throw new AccountCreateException("Customer does not exits ... ");
47          }
48
49          Account account = new Account(accountSequence.getNext(), customer, currency, new Money(0.0d));
50          customer.addAccount(account); //pridedam saskaita vartotojui
51          return account;
52      }
53
54      @Override
55      public Operation transferMoney(Account debitAccount, Account creditAcount, Money debitAmount) {
56          if (debitAccount.getBalance().isLessThan(debitAmount)) { //palyginam ar saskaita nera mazesne uz
    paimama suma ("isLessThan' yra is jar failo)
57              throw new InsufficientFundsException("You are bad customer ... ");
58          }
```

```
59
60          debitAccount.setBalance(debitAccount.getBalance().substract(debitAmount)); //is saskaitos atimam
    paimamsuma (substract reiskia atimti)
61          //negalim tiesiog prideti debitAmount, nes gali buti operacija tarp skirtingu valiutu, todel
    pradzioj reikia konvertuoti:
62          Money creditAmount = currencyConverter.convert(debitAccount.getCurrency(), creditAcount.getCurrency(
    ), debitAmount );
63          creditAcount.setBalance(creditAcount.getBalance().add(creditAmount));
64          return new Operation(operationSequence.getNext(), debitAccount, creditAcount, debitAmount);
65      }
66
67      @Override
68      public Money getBalance(Currency currency) {
69          Money balance = new Money (0.0d); //sukuriam balansa su nuliu valiutos
70
71          //kadangi reikia balanso ir visu saskaitu ir tik eurais, tai iteruojam per visus vartotojus ir
    iteruojam per visas vartotojo saskaitas:
72          //jei visu saskaitu valiuta Eur, tai pridedam iskart, jei ne - tai pirma konvertuojam ir tada
    pridedam
73          for(Customer customer : customers) {
74              for (Account account : customer.getAccounts()) {
75                  if (account.getCurrency().equals(currency)) {
76                      balance = balance.add(account.getBalance());
77                  } else {
78                      balance = balance.add(currencyConverter.convert(account.getCurrency(), currency, account
    .getBalance()));
79                  }
80              }
81          }
82          return balance;
83      }
84  }
85
```