

```

1 import lt.itakademija.exam.*;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 public class BankImpl implements Bank {
7
8     private final CurrencyConverter currencyConverter; //deklaruojam valiutos keitykla
9     private final List<Customer> customers; //deklaruojam lista kad butu kur deti vartotojus, sukuriamus
        pagal metoda 'createCustomer'
10
11     //trys generatoriai eiliskumui-unikalumui:
12     private SequenceGenerator customerSequence = new SequenceGenerator(); //sukuriam generatoriu kad
        vartotojas turetu eiliskuma
13     private SequenceGenerator accountSequence = new SequenceGenerator(); //sukuriam generatoriu saskaitai
        kita valiuta
14     private SequenceGenerator operationSequence = new SequenceGenerator(); //sukuriam generatoriu operaciju
        eiliskumui
15
16
17     public BankImpl(CurrencyConverter currencyConverter) { //'Generate --> Konstruktor'
18         this.currencyConverter = currencyConverter;
19         this.customers = new ArrayList<>(); //rankom (buvo this.customers = customers)
20     }
21
22     @Override // visi override generuojami, o paskui koreguojami rankomis
23     public Customer createCustomer(PersonCode personCode, PersonName personName) {
24         if (personCode == null || personName == null) { //cia tik patikrinimas ar vartotojas turi duomenis (
            koda ir varda)
25             throw new NullPointerException();
26         }
27
28         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 exists ... ");
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 creditAccount, 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         debitAccount.setBalance(debitAccount.getBalance().subtract(debitAmount)); //is saskaitos atimam
60         paimamsuma (subtract reiskia atimti)
61         //negalim tiesiog prideti debitAmount, nes gali buti operacija tarp skirtingu valiutu, todel
62         pradzioj reikia konvertuoti:
        Money creditAmount = currencyConverter.convert(debitAccount.getCurrency(), creditAccount.getCurrency(
        ), debitAmount );
63         creditAccount.setBalance(creditAccount.getBalance().add(creditAmount));
64         return new Operation(operationSequence.getNext(), debitAccount, creditAccount, 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 euras, tai iteruojam per visus vartotojus ir
72         iteruojam per visas vartotojo saskaitas:
        //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
79                         .getBalance()));
79                 }
80             }
81         }
82         return balance;
83     }
84 }
85

```