

Autoren: Marius Birk
Pieter Vogt
Tutor: Florian Brandt

Abgabe: 12.05.2020, 12:00 Uhr

Smileys:

A1	A2	Σ

Objektorientierte Modellierung und Programmierung

Abgabe Uebungsblatt Nr.03

(Alle allgemeinen Definitionen aus der Vorlesung haben in diesem Dokument bestand, es sei den sie erhalten eine explizit andere Definition.)

Aufgabe 1

a)

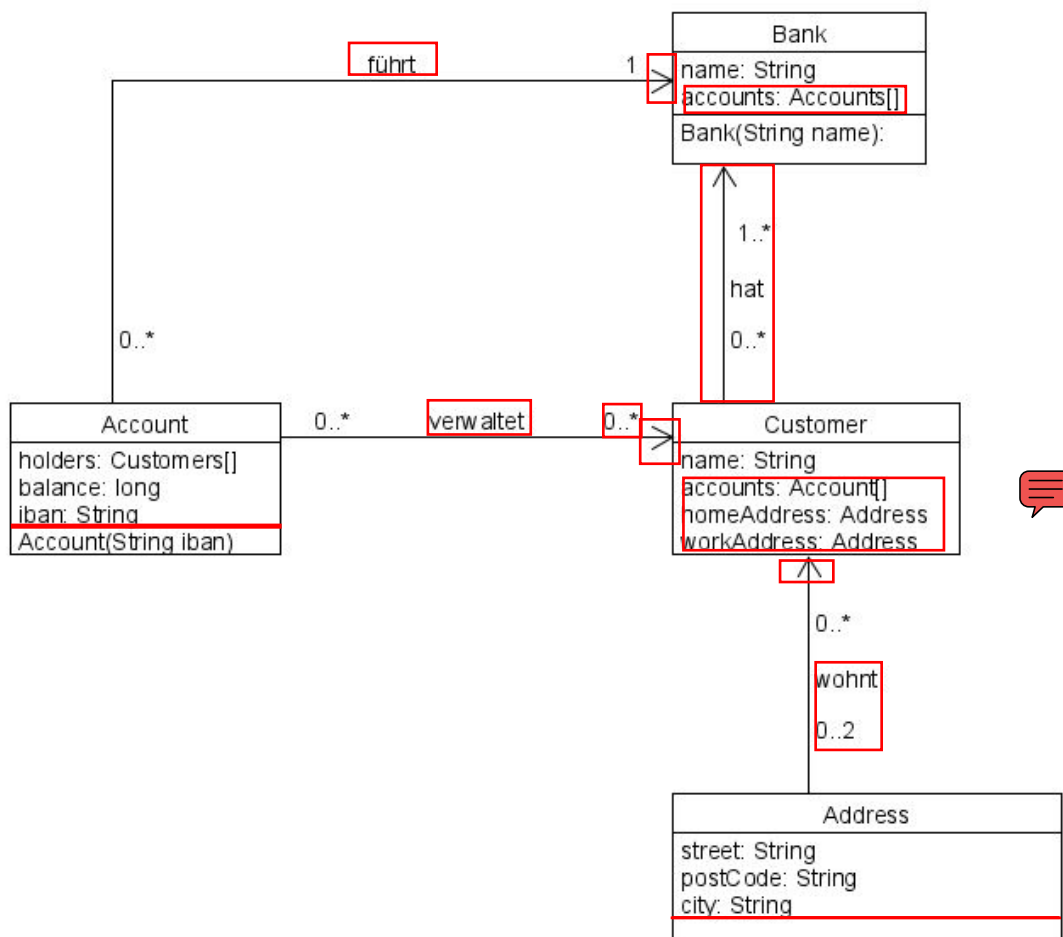


Abbildung 1: Klassendiagramm

a) 1/3

b)

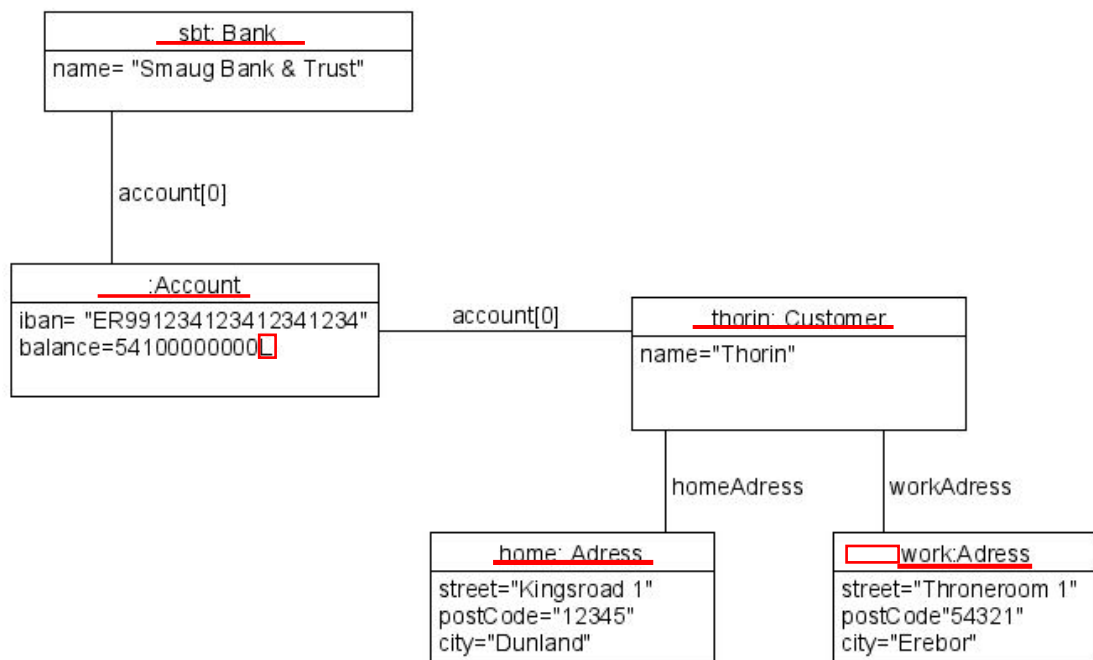


Abbildung 2: Objektdiagramm

b) 1.5/2

c)

Account Klasse

```

1 public class Account {
2
3     private Customer[] holders;
4     private long balance;
5     private String iban;
6
7     public Account(String iban) { this.iban = iban; }
8
9     public Customer[] getHolders() { return holders; }
10
11    public void setHolders(Customer[] holders) { this.holders =
        holders; }
12
13    public long getBalance() { return balance; }
14
15    public void setBalance(long balance) { this.balance =
        balance; }
16
17    public String getIban() { return iban; }
18

```

19 }

Adress Klasse

```
1 public class Address {
2
3     private String street;
4     private String postCode;
5     private String city;
6
7     public String getStreet() { return street; }
8
9     public void setStreet(String street) { this.street = street;
10         }
11
12     public String getPostCode() { return postCode; }
13
14     public void setPostCode(String postCode) { this.postCode =
15         postCode; }
16
17     public String getCity() { return city; }
18
19     public void setCity(String city) { this.city = city; }
20 }
```

Bank Klasse

```
1 public class Bank {
2
3     private String name;
4     private Account[] accounts;
5
6     public Bank(String name) { this.name = name; }
7
8     public String getName() { return name; }
9
10    public void setName(String name) { this.name = name; }
11
12    public Account[] getAccounts() { return accounts; }
13
14    public void setAccounts(Account[] accounts) { this.accounts
15        = accounts; }
16 }
```

Banking Klasse

```
1 public class Banking {
2
3     public static void main(String[] args) {
4         Bank sbt = new Bank("Smaug Bank & Trust");
5         sbt.setAccounts(new Account[1]);
6     }
7 }
```

```
6     sbt.getAccounts()[0] = new Account("ER99123412341234123412"
7         );
8     sbt.getAccounts()[0].setBalance(54100000000L);
9     Customer thorin = new Customer();
10    thorin.setAccounts(new Account[1]);
11    thorin.getAccounts()[0] = sbt.getAccounts()[0];
12    thorin.setName("Thorin");
13    Address home = new Address();
14    home.setStreet("Kingsroad 1");
15    home.setPostCode("12345");
16    home.setCity("Dunland");
17    thorin.setHomeAddress(home);
18    Address work = new Address();
19    work.setStreet("Throneroom 1");
20    work.setPostCode("54321");
21    work.setCity("Erebor");
22    thorin.setWorkAddress(work);
23    sbt.getAccounts()[0].setHolders(new Customer[] { thorin });
24 }
25 }
```

Customer Klasse

```
1 public class Customer extends Person{
2     private Account[] accounts;
3
4     public Account[] getAccounts() { return accounts; }
5
6     public void setAccounts(Account[] accounts) { this.accounts
7         = accounts; }
8 }
```

FinancialAdvisor Klasse

```
1 public class FinancialAdvisor extends Person{
2     private Account[] supervised;
3 }
```

HomeAdress Klasse

```
1 public class HomeAddress extends Address{
2     private String poBoxCode;
3     private String poBoxCity;
4
5     public String getPoBoxCode() {
6         return poBoxCode;
7     }
8
9     public void setPoBoxCode(String poBoxCode) {
10        this.poBoxCode = poBoxCode;
11    }
12 }
```

```
13     public String getPoBoxCity() {
14         return poBoxCity;
15     }
16
17     public void setPoBoxCity(String poBoxCity) {
18         this.poBoxCity = poBoxCity;
19     }
20 }
```

Person Klasse

```
1  public class Person {
2      private String name;
3      private Address homeAddress;
4      private Address workAddress;
5
6      public String getName() {
7          return name;
8      }
9
10     public void setName(String name) {
11         this.name = name;
12     }
13
14     public Address getHomeAddress() {
15         return homeAddress;
16     }
17
18     public void setHomeAddress(Address homeAddress) {
19         this.homeAddress = homeAddress;
20     }
21
22     public Address getWorkAddress() {
23         return workAddress;
24     }
25
26     public void setWorkAddress(Address workAddress) {
27         this.workAddress = workAddress;
28     }
29 }
```

WorkAddress Klasse

```
1  public class WorkAddress extends Address{
2      private String companyName;
3
4      public String getCompanyName() {
5          return companyName;
6      }
7
8      public void setCompanyName(String companyName) {
9          this.companyName = companyName;
10     }
```

```

10     }
11 }

```



c) 3/5

Aufgabe 2

1) 5.5/10

Die VersatileLinked List:

```

1  public class VersatileLinkedList extends LinkedListStringList {
2
3      public void add(int i) {
4          super.add(Integer.toString(i));
5      }
6
7      public void add(boolean b) {
8          if (b == true) {
9              super.add("yes");
10             } else {
11                 super.add("no");
12             }
13         }
14
15         public void add(LinkedListStringList list) {
16             for (int i = 0; i < list.size(); i++) {
17                 super.add(list.get(i));
18             }
19         }
20
21         public void add(LinkedListStringList list, int start, int end)
22         {
23             if (start > end || start < 0 || end > list.size()) {
24                 return;
25             } else {
26                 for (int i = start; i < end; i++) {
27                     super.add(list.get(i));
28                 }
29             }
30
31         public VersatileLinkedList reverse() {
32             VersatileLinkedList temp = new VersatileLinkedList();
33             for (int i = this.size() - 1; i >= 0; i--) {
34                 temp.add(this.get(i));
35             }
36             return temp;
37         }
38
39         public boolean equals(VersatileLinkedList list) {
40             if (this.size() != list.size()) {
41                 return false;

```

a) 1/1

b) 1/1

c) 1/1

d) 2/2

e) 2/2

```
42     } else {
43         for (int i = 0; i < this.size(); i++) {
44             if (this.get(i).equals(list.get(i))) {
45                 i++;
46             } else {
47                 return false;
48             }
49         }
50         return true;
51     }
52 }
53
54 public void print() {
55     for (int i = 0; i < this.size(); i++) {
56         System.out.println(this.get(i));
57     }
58 }
59 }
```

f) 0.5/1

Die Main Methode zum testen:

```
1 public class Main {
2     public static void main(String[] args) {
3
4         VersatileLinkedList ListA = new VersatileLinkedList();
5         VersatileLinkedList ListB = new VersatileLinkedList();
6         VersatileLinkedList ListC = new VersatileLinkedList();
7
8         ListA.add(3);
9         ListA.add(true);
10        ListA.add("Hunter");
11        ListA.add(7);
12        ListA.add(9);
13
14        ListB.add("Dorms");
15        ListB.add(false);
16        ListB.add("Hunter");
17        ListB.add("Vepr");
18        ListB.add(9812);
19
20        ListC = ListA;
21
22        //tests
23
24        System.out.println(ListA.equals(ListB));
25        System.out.println(ListA.equals(ListC));
26
27        //ListA.add(ListB,2,3);
28        //ListC = ListA.reverse();
29        //ListA.print();
```

```
30      //ListB.print();
31      //ListC.print();
32
33    }
34 }
```



g) 1.5/2

2) 9/10