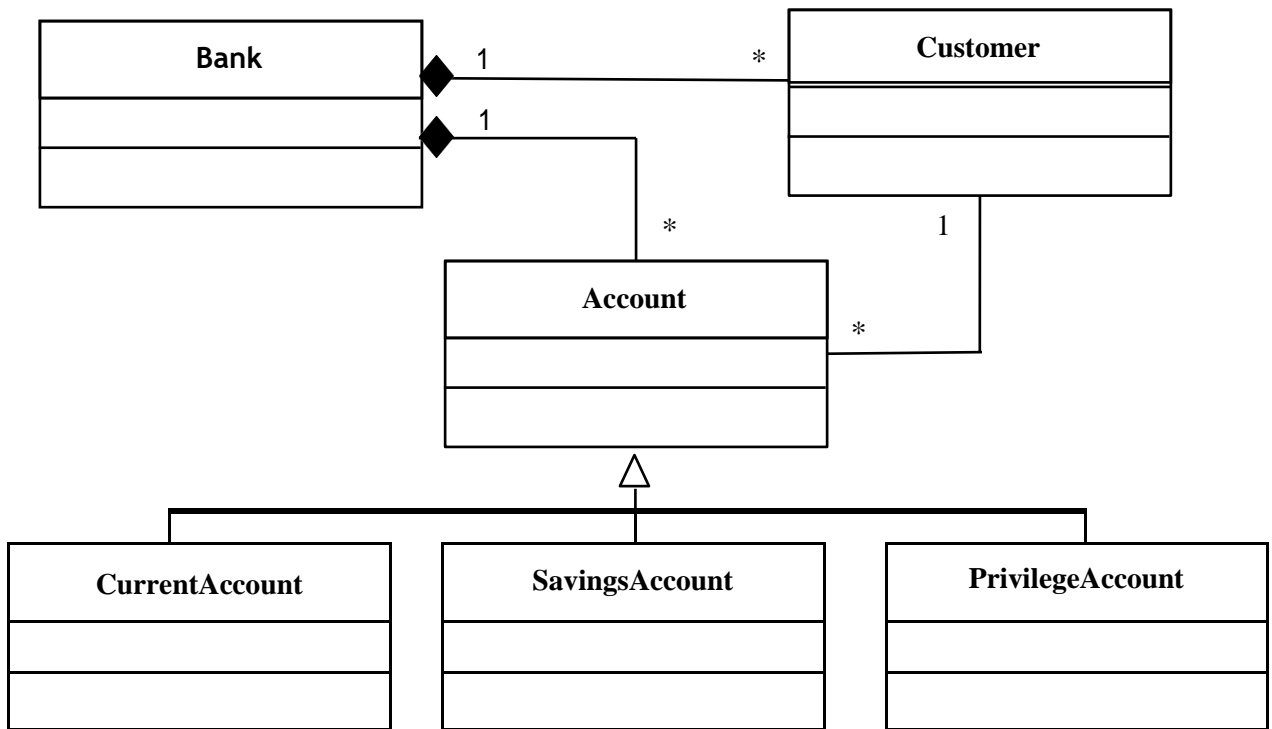


Задача: Банкова система

Да се напише програма, която моделира банкова система и операциите, които се извършват в нея. За целта е необходимо да се реализират класовете, показани на следната UML клас диаграма:



1. Да се създаде клас **Customer** (потребител)

Член-данни:

- **id** – идентификатор на потребителя
- **name** – име на потребителя
- **address** – адрес на потребителя

Методи:

- конструктор с параметри
- селектори: **getId**, **getName**, **getAddress**
- **display** – извежда информация за потребителя
- допълнителни методи, ако е необходимо

2. Да се създаде абстрактен клас **Account** (банкова сметка)

Член-данни:

- **iban** – номер на банкова сметка (IBAN)
- **ownerId** – идентификатор на потребителя, собственик на банковата сметка
- **amount** – налична сума в сметката

Методи:

- конструктор с параметри
- чисто виртуален метод **deposit** – за добавяне на сума към банковата сметка
- чисто виртуален метод **withdraw** – за изтегляне на сума от банковата сметка, ако това е възможно (връща true или false)
- чисто виртуален метод **display** – за извеждане на информация за сметката
- **getBalance** – връща наличната сума в сметката
- допълнителни методи, ако е необходимо

3. Да се създадат следните производни класове

3.1. Клас **CurrentAccount** (текуща сметка)

Класът **CurrentAccount** наследява **Account** и реализира методите:

- **deposit** – за добавяне на сума към банковата сметка
- **withdraw** – за изтегляне на сума от банковата сметка. Ако в сметката има по-малко пари от исканата сума, връща false, иначе – намалява сумата на сметката с исканата сума и връща true
- **display** – за извеждане на информация за сметката – вид на сметка, IBAN, номер на потребител, баланс
- допълнителни методи, ако е необходимо

3.2. Клас **SavingsAccount** (спестовна сметка)

Класът **SavingsAccount** наследява **Account** като го допълва с

- **interestRate** – годишен лихвен процент

и реализира методите:

- **getInterestRate** – връща годишния лихвен процент
- **deposit** – за добавяне на сума към банковата сметка
- **withdraw** – за изтегляне на сума от банковата сметка. Ако в сметката има по-малко пари от исканата сума, връща false, иначе – намалява сумата на сметката с исканата сума и връща true
- **display** – за извеждане на информация за сметката – вид на сметка, IBAN, номер на потребител, годишния лихвен процент, баланс
- допълнителни методи, ако е необходимо

3.3. Клас **PivilegeAccount** (привилегирована сметка)

Класът **PivilegeAccount** наследява **Account** като го допълва с

- **overdraft** – позволен овърдрафт (сума превишаваща кредита)

и реализира методите:

- **getOverdraft** – връща позволения овърдрафт
- **deposit** – за добавяне на сума към банковата сметка
- **withdraw** – за изтегляне на сума от банковата сметка. Ако наличната сума в сметката + позволения овърдрафт е по-малко от исканата сума, връща false, иначе – намалява сумата на сметката с исканата сума и връща true (в този случай може да се получи отрицателно число за баланса)

- **display** – за извеждане на информация за сметката – вид на сметка, IBAN, номер на потребител, позволен овърдрафт, баланс
- допълнителни методи, ако е необходимо

4. Да се създаде клас **Bank** (банка)

Член-данни:

- **name** – име на банката
- **address** – адрес на банката
- **customers** – списък от потребители (изберете подходящ начин за представяне)
- **accounts** – списък от банкови сметки (изберете подходящ начин за представяне)

Методи:

- конструктор с параметри
- селектори: **getName**, **getAddress**
- **addCustomer** с параметри **customerId**, **name** и **address** – ако не съществува потребител с идентификатор **customerId**, създава нов потребител и го добавя към списъка с потребителите
- **listCustomers** – извежда списък с потребителите
- **deleteCustomer** с параметър **customerId** – изтрива потребител с идентификатор **customerId** от списъка с потребители, както и всички негови сметки
- **addAccount** с параметри **accountType**, **iban**, **ownerId**, **amount** – създава нова сметка от съответния тип и я добавя към списъка с банкови сметки. Ако не съществува потребител с идентификатор **ownerId** или съществува банкова сметка със същия IBAN – извежда съобщение за грешка.
- **deleteAccount** с параметър **iban** – изтрива сметка с съответен IBAN
- **listAccounts** – извежда информация за всички сметки в банката
- **listCustomerAccount** с параметър **customerId** – извежда информация за банковите сметки на потребител с идентификатор **customerId**
- **transfer** с параметри **fromIBAN**, **toIBAN** и **amount** – извършва банков превод на сума **amount** от банкова сметка с идентификатор **fromIBAN** към банкова сметка с идентификатор **toIBAN**, ако е възможно
- **display** – извежда информация за банката и колко потребителя и сметки има
- допълнителни методи, ако е необходимо

5. Да се напише главна програма

В главната програма да се създаде банка и да се реализира следното меню:

```

1  List customers
2  Add new customer
3  Delete customer
4  List all accounts
5  List customer accounts
6  Add new account
7  Delete account
8  Withdraw from account
9  Deposit to account
10 Transfer
11 Display info for the bank
12 Quit

```