Praktikum 1

Voraussetzungen:

- Entwicklungsumgebung (IDE bzw. Editor) für Java
- Java-Installation (JDK), empfohlen mind. Version 14 (getestet mit Java 17)

Das im OOS-Praktikum verwendete Beispiel soll einen stark vereinfachten Kontext einer Bank darstellen (Umgang mit Konten, Ein- und Auszahlungen und Überweisungen). Die Funktionalität wird hierfür in jedem Prak-tikumsversuch Schritt für Schritt erweitert.
In diesem Versuch werden Sie zwei Basisklassen erstellen und die ersten grundlegenden Eigenschaften der Objektorientierung an praktischen Beispielen erlernen.

Aufgabe 1 (Erstellung eines Java-Projekts und Implementierung von zwei Basisklassen):
Erstellen Sie zunächst ein Java-Projekt in Ihrer gewählten Entwicklungsumgebung. Legen Sie anschließend ein Java-Package *bank* an, das die beiden folgenden (Java-)Basisklassen beinhalten soll:
Die erste Basisklasse *Payment* soll Ein- und Auszahlungen repräsentieren; die zweite Basisklasse *Transfer* soll im Kontext von Überweisungen verwendet werden.
Diese beiden Klassen haben grundsätzlich viele Gemeinsamkeiten, insbesondere sollen drei der jeweils fünf Klassenattribute identisch sein:

1. *date* (Typ: String): Dieses Attribut soll den Zeitpunkt einer Ein- oder Auszahlung bzw. einer Überweisung darstellen. Das Datumsformat soll als *DD.MM.YYYY* angegeben werden (dies muss *nicht* programmatisch überprüft werden).
2. *amount* (Typ: double): Dieses Attribut soll die Geldmenge einer Ein- oder Auszahlung bzw. einer Überweisung darstellen.
   Wichtig: Bei *Payment*-Objekten kann sowohl ein negativer (-> Auszahlung) als auch ein positiver Wert (-> Einzahlung) angegeben werden; bei *Transfer*-Objekten sind lediglich positive Werte als Eingabe erlaubt (dies muss programmatisch überprüft werden. Eine fehlerhafte Eingabe soll auf die Konsole geschrieben werden, *System.out*).
3. *description* (Typ: String): Dieses Attribut erlaubt eine zusätzliche Beschreibung des Vorgangs.

Überlegen Sie generell bei allen Klassenattributen, welche Sichtbarkeit sinnvoll wäre (private, public, …).

Des Weiteren soll die Basisklasse *Payment* folgende Klassenattribute enthalten:

1. *incomingInterest* (Typ: double): Dieses Attribut gibt die Zinsen (positiver Wert in Prozent, 0 bis 1; dies muss programmatisch überprüft werden. Eine fehlerhafte Eingabe soll auf die Konsole geschrieben werden, *System.out*) an, die bei einer Einzahlung („Deposit“) anfallen.
2. *outgoingInterest* (Typ: double): Dieses Attribut gibt die Zinsen (positiver Wert in Prozent, 0 bis 1; dies muss programmatisch überprüft werden. Eine fehlerhafte Eingabe soll auf die Konsole geschrieben werden, *System.out*) an, die bei einer Auszahlung („Withdrawal“) anfallen.

Transfer soll hingegen folgende Klassenattribute enthalten:

1. *sender* (Typ: String): Dieses Attribut gibt an, welcher Akteur die Geldmenge, die in *amount* angegeben wurde, überwiesen hat.
2. *recipient* (Typ: String): Dieses Attribut gibt an, welcher Akteur die Geldmenge, die in *amount* angegeben wurde, überwiesen bekommen hat.

Implementieren Sie für alle oben erwähnten Klassenattribute Getter- und Settermethoden und verwenden Sie diese!

Im nächsten Schritt sollen Sie jeweils drei Konstruktoren für die Klasse *Payment* und *Transfer* implementieren:

1. Konstruktor für die Attribute *date*, *amount* und *description.*
2. Konstruktor für alle Attribute. Vermeiden Sie redundanten Code, indem Sie den Konstruktor aus 1. wiederverwenden.
3. *Copy-Konstruktor*. Recherchieren Sie hierfür selbstständig, was ein *Copy-Konstruktor* ist und wie dieser zu implementieren ist.

Implementieren Sie im letzten Schritt dieser Aufgabe eine parameterlose public-Methode *printObject*(), die den Inhalt aller Klassenattribute auf der Konsole (*System.out*) ausgibt.

Aufgabe 2 (Dokumentation mit Hilfe von Java-Kommentaren):
In dieser Aufgabe sollen Sie alle Klassen, Methoden (inkl. Konstruktoren) und Klassenattribute mit einfachen Java-Kommentaren dokumentieren. Geben Sie hierfür jeweils sinnvolle Beschreibungen an, so dass der Programmcode auch für Personen, die den Programmcode noch nicht kennen bzw. noch nicht eingearbeitet sind, möglichst leicht verständlich ist.

Aufgabe 3 (Testen der Funktionalität):
In der letzten Aufgabe soll die in Aufgabe 1 erstellte Funktionalität mit Hilfe einer *main*-Methode getestet werden.
Erstellen Sie hierfür eine Klasse *Main,* die außerhalb des Java-Package *bank* liegen soll. Die Klasse *Main* soll eine *main*-Methode enthalten und als Einstiegspunkt für folgende Tests fungieren:
Erzeugen Sie Objekte der Klassen *Payment* und *Transfer* und verwenden Sie dabei alle zur Verfügung stehenden Konstruktoren (inkl. *Copy-Konstruktor*). Überlegen Sie hierbei, wie Sie sicherstellen können, dass alle Konstruktoren richtig implementiert wurden.
Zusätzlich sollen die Fehlerfälle bei der Validierung von bestimmten Klassenattributen getestet werden (s. Aufgabe 1).