

# Softwaretechnologie für Natürlichsprachliche Systeme

Sommersemester 2010, Prof. Dr. Udo Hahn, Erik Fäßler  
Übungsblatt 5 vom 20.05.2010  
Abgabe bis Zum nächsten Veranstaltungstermin vorzeigen können

---

## Aufgabe 1 : Implementierung eines Klassensystems

In der letzten Übung sollten Sie einige interagierende Klassen skizzieren. Implementieren Sie diese Klassen nun, so dass mindestens folgende Aufgabe gelöst werden kann:

Stellen Sie eine Situation dar, in der eine Person namens Peter kein Geld mehr in der Tasche hat. Er lässt sich seinen Kontostand anzeigen und hebt anschließend über den Geldautomaten 100€ von einem Konto ab. Anschließend sieht er in seine Geldbörse und gibt die Summe seines Geldes auf dem Bildschirm aus.

Außerdem wird das Klassengeflecht erweitert. Es sieht nun so aus:

1. Eine Klasse `Cash`, die einen Bargeldvorrat repräsentiert. Sie sollte enthalten
  - Fünf Attribute `amount100`, `amount50`, `amount20`, `amount10` und `amount5` vom Typen `int`, die die Anzahl der jeweiligen Geldscheinart beinhalten. *Hinweis*: Wenn Sie es sich zutrauen, implementieren Sie den Bargeldvorrat eleganter durch ein Objekt der Java-Klasse `HashMap`.
  - Eine Methode `getCashAmount`, die die Summe der vorhandenen Geldwerte zurück gibt.
  - Eine Methode `getCash`, der als Parameter übergeben wird, welche Art der Banknote und wie viele davon entnommen werden sollen (*Hinweis* z.B. wieder als `HashMap`, das ist aber kein Muss). Die Methode soll ein `Cash`-Objekt zurück geben, das den entnommenen Betrag scheinergerecht repräsentiert.
  - Eine Methode `addCash`, die ein `Cash`-Objekt entgegen nimmt und dessen Wert dem eigenen Bargeldvorrat scheinergerecht hinzufügt.
2. Eine Klasse `Account`, die ein Konto repräsentiert. Sie sollte enthalten
  - Ein Attribut `balance` des Typs `double` für den Kontostand.
  - Methoden `getBalance`, `deposit` und `withdraw`, um den Kontostand abzufragen, Geld einzuzahlen bzw. welches abzuheben.
3. Eine Klasse `Atm`, die Kommandos von einem Kontoinhaber entgegen nimmt und als Zugriffsschnittstelle zu einem Konto dient. Sie sollte enthalten
  - Ein Attribut `cash` vom Typ `Cash`.
  - Ein Attribut `account` vom Typ `Account`.
  - Methoden `withdrawMoney` und `getBalance`, die von einem Benutzer aufrufbar sein sollen und mit einem `Account`-Objekt kommunizieren, um den entsprechenden Prozess durchzuführen.
  - Eine Methode `getMoneyAmount`, die den Gesamtwert aller im Automaten vorhandenen Bargeldmittel zurück gibt.
4. Eine Klasse `Person`, die mit dem Geldautomaten (bzw. mit der Klasse `Atm`) interagiert. Sie sollte enthalten
  - Ein Attribut `name` vom Typ `String`.
  - Ein Attribut `pin` vom Typ `int`.
  - Ein Attribut `cash` vom Typ `Cash`.
  - Eine Methode `getName`, die `name` zurück gibt.

Zusatzaufgabe: Programmieren Sie den Geldautomaten so, dass er möglichst große (und damit auch möglichst wenige) Scheine zurück gibt!

*Anmerkungen:*

- Sie sind natürlich frei, weitere Attribute oder Methoden hinzuzufügen, die Ihnen sinnvoll erscheinen.
- Achten Sie darauf, welche Attribute und Methoden nur für die Klasse selbst sichtbar sein sollen (Modifier `private`) und welche öffentlich sein sollen (Modifier `public`).
- Jede Klasse soll `public` sein und muss entsprechend in einer eigenen `.java`-Datei abgespeichert werden. Legen Sie alle `.java`-Dateien und die resultierenden `.class`-Dateien in das gleiche Verzeichnis, damit Sie eine Klasse aus der anderen heraus verwenden können.