Assignment 2

Bearbeiten Sie die nachfolgende Aufgabe und reichen Sie Ihre Lösung elektronisch bis zum 23. Dezember 2008, 10:00 Uhr ein. Senden Sie Ihre Lösung per E-Mail an die Adresse stephan.aier@unisg.ch (als **ZIP-Archiv** mit allen Teillösungen und benennen Sie das Archiv nach dem Schema **`A2-Nachname-Vorname.ZIP`**, ja, es soll genau eine Datei nach diesem Schema benannt abgegeben werden und sie soll als .zip komprimiert werden – nicht .rar, nicht .tar, nicht .tar.gz usw. – danke).

Für jeden angefangenen Tag Verzögerung in der Abgabe der Lösung werden pauschal 20% der erreichten Punkte abgezogen.

Aufgabe: Bank2 (30 Punkte)

Erstellen Sie ein neues Projekt *Bank2*, das verschiedene Bankgeschäfte durchführt.

a) Erstellen Sie zunächst die Klasse *Konto* (4 Punkte):
   - Die Datenfelder *vorname* und *nachname* speichern Vor- und Nachnamen des Kontoinhabers, *kontonummer* speichert die Kontonummer (Heinweis: Kontonummern können sowohl Zahlen als auch Buchstaben enthalten). Sie sollen durch Eingabeparameter im Konstruktor initialisiert werden.
   - Das Datenfeld *kontostand* enthält den Saldo des Kontos in Schweizer Franken. Es wird im Konstruktor mit 0.00 initialisiert. Hinweis: Erläuterungen zu Datentypen finden Sie in Anhang B des Buches.
   - Die Methode *einzahlen* ermöglicht die Einzahlung eines beliebigen Betrags in Schweizer Franken auf das Konto.
   - Die Methode *abheben* ermöglicht das Abheben eines beliebigen Betrags in Schweizer Franken.
   - Anmerkung: Es gibt hier im Gegensatz zu Assignment 1 keine Kreditlimite oder Geheimnummern.

b) Erstellen Sie eine Klasse *Kontoverwaltung* (8 Punkte).
   - Diese Klasse soll eine Sammlung einer beliebigen Anzahl von Konten enthalten. In der Sammlung soll über den Schlüssel *kontonummer* auf das jeweils zugehörige Konto zugegriffen werden können.
   - Mit der Methode *kontoAnlegen(<hier passende Parameter einfügen>)* soll es möglich sein, ein neues Konto zu erzeugen. Neu angelegte Konten müssen immer in die Sammlung aller Konten eingefügt werden.

c) Erweiterung der Klassen *Konto* und *Kontoverwaltung* (10 Punkte):
   - Mit der Methode *kontoSperren()* soll es möglich sein, alle Konten, die einen negativen *kontostand* aufweisen mit Hilfe eines Datenfelds als gesperrt zu markieren.
   - Von gesperrten Konten soll kein Geld abgehoben werden können.

- Mit der Methode *kontenAuflisten()* soll es möglich sein, die Attribute aller Konten auf der Konsole auszugeben.
- Mit der Methode *kontoinhaberAuflisten()* soll es möglich sein, alle Kontoinhaber (identifizierbar durch die Kombination von Vor- und Nachname) auf der Konsole auszugeben. Achten Sie darauf, dass Kontoinhaber, die mehrere Konten besitzen nur einmal aufgelistet werden.

d) Spezialisierung der Klasse *Konto* (8 Punkte):

- Neben den „normalen Konten" soll es auch ein spezielles G*eschaeftskonto* geben. Ein *Geschaeftskonto* unterscheidet sich vom normalen Konto dadurch, dass es ein weiteres Datenfeld *firma* speichern kann. Wählen Sie eine Implementierung, die so wenig Code wie möglich dupliziert.
- Ein *Geschaeftskonto* wird durch die Methode *kontoSperren()* erst gesperrt, wenn der *kontostand* unter ein für jedes *Geschaeftskonto* bei der Eröffnung festgelegtes *limit* fällt.
- Mit der Methode *geschäftskontoAnlegen(<hier passende Parameter einfügen>)* soll es möglich sein, ein neues Konto zu erzeugen und gleichzeitig das *limit* für dieses G*eschaeftskonto* festzulegen. Neu angelegte Geschäftskonten müssen immer in die Sammlung aller Konten eingefügt werden.

Zusatzaufgabe (6 Punkte)

Erweitern Sie das obige Projekt so, dass bei der Ausführung von *kontenAuflisten()* im Falle eines Geschäftskontos zusätzlich auch die *firma* sowie das *limit* aufgelistet werden. Ebenso soll durch *kontoinhaberAuflisten()* die *firma* zusätzlich gelistet werden. Beachten Sie aber, dass auch wirklich alle Firmen eines ggf. gleichen Kontoinhabers aufgelistet werden.

Hinweis zu allen Aufgaben:

- Die Aufgaben bauen teilweise aufeinander auf. Es werden für jeden Lösungsschritt (Teilaufgabe) Punkte vergeben. Sofern nicht anders angegeben, genügt es zum Erreichen der vollen Punktzahl, ein einziges Programm abzugeben, das die komplette Lösung der Aufgabe enthält.
- Beachten Sie bitte, dass bei der Bewertung die Hälfte der Punkte für die Dokumentation der funktionierenden Teile des Sourcecodes anfällt, d.h. versehen Sie den Sourcecode mit Kommentaren in der Form, dass er nachvollzogen werden kann und `javadoc`-kompatibel ist. Orientieren Sie sich dabei an den Hinweisen aus der Vorlesung sowie an den Kommentaren der Übungsprojekte in den Vorlesungen. Die Kommentare sind multiplikativ, d.h. für nicht funktionierenden

Code gibt es auch bei Kommentaren keine Punkte und umgekehrt auch für funktionierenden Code ohne Dokumentation gibt es keine Punkte.

- Die maximal erreichbare Punktzahl ist 30. Sollten Sie jedoch in der Aufgabe nicht die volle Punktzahl erreichen, so kann dies ggf. durch Punkte der Zusatzaufgabe ausgeglichen werden.