Universität Augsburg
Institut für Informatik

Übung zur Vorlesung Informatik 2 SoSe 2023
Fakultät für Angewandte Informatik
Institut für Informatik
Dr. M. Frieb, Dr. D. Müller, V. Le Claire, M. Steinberger 26.05.2023

Übungsblatt 6

Abgabe: 12.06.2023, 09:00 Uhr (via Digicampus an den Tutor: .java-Dateien für Code, .uxf für UML, .pdf für alles andere)

- Dieses Übungsblatt muss im Team abgegeben werden (Einzelabgaben sind nicht erlaubt!).
- Die **Zeitangaben** geben zur Orientierung an, wie viel Zeit für eine Aufgabe später in der Klausur vorgesehen wäre; gehen Sie davon aus, dass Sie zum jetzigen Zeitpunkt wesentlich länger brauchen und die angegebene Zeit erst nach ausreichender Übung erreichen.

* leichte Aufgabe / ** mittelschwere Aufgabe / *** schwere Aufgabe

Aufgabe 26 + 27 *(Eigene Klasse entwerfen, 45 Minuten)*

In dieser Aufgabe sollen Sie eine eigene Klasse implementieren.

a) (**, 42 Minuten)
Implementieren Sie gemäß folgender Systembeschreibung eine eigene Klasse `BankAccount` zur Verwaltung von Bankkonten.

Bankkonten haben den Namen des Kontohalters, eine Kontonummer, ein Eröffnungsdatum sowie einen Kontowert. Der Name des Kontoinhabers ist eine Zeichenkette und muss aus mindestens einem Wort bestehen, das mit einem lateinischen Großbuchstaben beginnt, gefolgt von beliebig vielen lateinischen Kleinbuchstaben. Der Name des Kontoinhabers kann aus mehreren Worten, getrennt durch Leerzeichen bestehen. Der Name des Kontoinhabers kann jederzeit geändert und gelesen werden. Die Kontonummer ist eine Zeichenkette, die aus genau 12 *Großbuchstaben und Ziffern besteht, z.B.* `"A12B34C56D78"`*. Die Kontonummer wird beim Erstellen des Bankkontos festgelegt und kann danach nicht mehr geändert werden. Wird für den Namen oder die Kontonummer ein ungültiger Wert übergeben, so wird eine geeignete ungeprüfte Ausnahme geworfen. Das Eröffnungsdatum entspricht dem Tag, an dem das Bankkonto erstellt wurde und kann nicht geändert werden. Der Kontostand ist eine reelle Zahl und ist zu Beginn* 0*. Danach kann der Kontostand sowohl erhöht als auch verringert werden. Alle Attribute des Bankkontos sind öffentlich lesbar. Zwei Bankkonten sind genau dann gleich, wenn ihre Kontonummern gleich sind. Wählen Sie eine geeignete Darstellung des Bankkontos als Zeichenkette.*

Verwenden Sie für die Attribute die Namen `accountHolderName`, `accountNumber`, `openingDate` und `balance`.

Hinweis: Eine elegante Möglichkeit zur Überprüfung des Namens und der Kontonummer sind die in der Klasse `Pattern` beschriebenen regulären Ausdrücke zur Überprüfung von Zeichenketten. Benutzen Sie außerdem die Tatsache, dass Klammern in regulären Ausdrücken Teile des Ausdrucks gruppieren können: `"ab+"` matcht `"ab"` sowie `"abb"` usw., während `"(ab)+"` `"ab"`, `"abab"` usw. matcht. Es ist aber genauso möglich, die Überprüfungen ohne reguläre Ausdrücke zu machen.

b) (*, 3 Minuten)
Testen Sie Ihre Implementierung mit der im Digicampus zur Verfügung gestellten Klasse `A2627_main.java`.
Hinweis: Für diese Teilaufgabe ist keine Abgabe erforderlich. Die von Ihnen erstellte Klasse muss zusammen mit der gegebenen Programmklasse lauffähig sein.

Aufgabe 28 *(Eigene Datenklasse implementieren, 25 Minuten)*

In dieser Aufgabe sollen Sie eine eigene Klasse implementieren, die das Verhalten einer Bank darstellt.

a) (***, 21 Minuten)
Implementieren Sie eine eigene Datenklasse `Bank` nach folgender Systembeschreibung:
Eine Bank verwaltet Bankkonten, die einzigartige Kontennummern, einen Inhaber sowie einen Kontostand haben. Eine Bank kann neue Konten eröffnen lassen, dazu bekommt Sie den Namen des Inhabers sowie die gewünschte Kontonummer gegeben. Ist bereits ein Konto mit dieser Kontonummer bei der Bank angelegt worden, so wird eine `IllegalBankingException` *geworfen (Die Ausnahme wird in der nächsten Teilaufgabe implementiert). Die Bank stellt eine Möglichkeit zur Verfügung, die Summe aller Kontostände eines Inhabers zu erhalten. Wird versucht, die Summe aller Kontostände für einen Inhaber zu erhalten, der kein Konto in dieser Bank hat, so wird die Summe* 0 *zurückgegeben. Außerdem kann von bzw. auf einem Konto mit einer gegebenen Kontonummer Geld abgehoben bzw. eingezahlt werden. Wenn versucht wird, mehr als* 500 *Euro abzuheben oder einzuzahlen, oder eine nicht existente Kontonummer verwendet wird, so wird eine* `IllegalBankingException` *geworfen.*
Erzeugen Sie für die Berechnung der Summe aller Kontostände einen geeigneten Stream.

Hinweis: Verwenden Sie für die Verwaltung der Bankkonten sowohl eine geeignete Collection als auch eine Map. Die drei von der Klasse zur Verfügung gestellten Methoden nennen Sie `createBankAccount`, `getBalanceByHolder`, `addBalance`.

b) (*, 2 Minuten)
Implementieren Sie eine geprüfte Ausnahme `IllegalBankingException`, die das Setzen einer beliebigen Fehlernachricht erlaubt.

c) (*, 2 Minuten)
Testen Sie Ihre Implementierung mit der im Digicampus zur Verfügung gestellten Klasse `A28_main.java`.
Hinweis: Für diese Teilaufgabe ist keine Abgabe erforderlich. Die von Ihnen erstellte Klasse muss zusammen mit der gegebenen Programmklasse lauffähig sein.

Allgemeine Hinweise zur Implementierung von Fenstern:

- Fenster sind ausschließlich in AWT (Layout und Ereignisbehandlung) und Swing (grafische Komponenten) zu programmieren.
- Es sind immer genau die benötigten Importe anzugeben (keine Importe mit dem $*$-Operator).
- Auch wenn nicht explizit gefordert, sind Fenster immer als eigene Unterklassen einer API-Fensterklasse zu implementieren (siehe Vorlesungsskript und -beispiele).

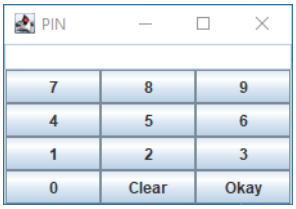
Aufgabe 29 ** *(Fenster ohne Ereignisbehandlung, 20 Minuten)*

In dieser Aufgabe sollen Sie die grundlegenden Elemente von Fenstern in Java näher kennen lernen.

a) (*, Eingabe mit Textfeld, 8 Minuten)
Implementieren Sie ein Hauptanwendungsfenster `A29a_main.java` nach folgenden Vorgaben:

- Es soll den Titel `Aufgabe29a` haben.
- Es soll im Norden von links nach rechts angeordnet die Beschriftung `Kontonummer` und ein aktiviertes einzeiliges Textfeld der Breite 20 anzeigen. Beschriftung und Textfeld sollen beide dieselbe Schrift mit Schriftgröße 20 haben.
- Es soll im Süden von links nach rechts angeordnet zwei Schaltflächen mit dem Text `Okay` und `Abbrechen` anzeigen.
- Der Nord- und der Südbereich sollen unterschiedliche Graufarben als Hintergrundfarbe haben.
- Durch Klicken auf das Fensterkreuz soll die Anwendung beendet werden.

b) (**, Visuelle Vorgabe, 12 Minuten)
Implementieren Sie ein Hauptanwendungsfenster `A29b_main.java`, das möglichst so wie in folgendem Bild aussieht:



Durch Klicken auf das Fensterkreuz soll die Anwendung beendet werden.

Hinweis: Die weiße Fläche im oberen Teil des Fensters soll dazu genutzt werden, die Eingabe anzuzeigen (Diese Funktionalität muss *nicht* implementiert werden).

Aufgabe 30 * *(Aufgabe zum Uni.Profi-Training)*

Das **fünfte UniProfi-Kapitel „Gefühle: Unterschätzte Verbündete“** (ca. 30 Minuten) ist in Digicampus **ab Montag, 29.05.2023** freigeschaltet. Klicken Sie sich rein und verbessern Sie Ihre Studierkompetenzen!
Für die Bearbeitung der Aufgaben in Kapitel 5 können Sie **0,5 Bonuspunkte** erhalten. Denken Sie außerdem daran, dass Sie bei mindestens 7 erfolgreich bearbeiteten UniProfi-Kapiteln (also 3,5 gesammelten Bonuspunkten) zusätzlich 0,5 Bonuspunkte erhalten. Die Einreichung der Aufgaben ist **bis Montag, 12.06. (9 Uhr)** möglich. Wenn Sie dem Kurs noch nicht beigetreten sind, können Sie dies nun nachholen: https://digicampus.uni-augsburg.de/dispatch.php/course/details?sem_id=bffff7d141569f9b1a3baf773286aa42&again=yes
(Passwort: BJf7S73uNX)

Viel Erfolg und viel Spaß!