



Aufgabe 0.6 (1 Punkt)

Geben Sie das Lösungswort des Quiz aus der Lernnotiz an.

Aufgabe 18 (2 Punkte)

Beantworten Sie folgende Fragen und Aufgaben schriftlich in ganzen Sätzen, Stichpunkten oder/und Graphiken.

- Gegeben sei eine Klasse K , was bedeutet es, wenn in C++ eine Methode die Form $\text{meth}(K\ k)$ hat, was wenn sie die Form $\text{meth}(K^* k)$ hat.
- Angenommen Sie müssen eine Liste mit Strings selbst programmieren, wie sieht ihr Klassendiagramm dazu aus, wenn es die Klasse `String`, aber keine Klassen oder Interfaces mit Listen (oder Arrays) gibt, Sie die Liste also selbst „bauen“ müssen?
- Erstellen Sie ein (Meta-)Klassendiagramm für Informationen in Klassendiagrammen, das u. a. folgende Informationen enthält. Dabei soll es z. B. eine Klasse namens `Klasse` geben, die die Objektvariable `klassenname` hat. Eine Klasse hat dann u. a. eine Sammlung von Methoden, die einen Namen und einen Rückgabetypen und eine Liste von Parametern haben. Vervollständigen Sie diese Überlegungen, dabei können Sie auf die Modellierung von Vererbung verzichten und davon ausgehen, dass jede Klasse eine Objektvariable `id` zur eindeutigen Identifikation der Objekte hat. Es werden keine Verwaltungsklassen benötigt, weiterhin sollen alle Variablen als Typ eine echte Klasse haben, damit müssen `int`, `char`, ... nicht berücksichtigt werden. Konstruktoren sollen als normale Methoden modelliert werden.

Aufgabe 19 (5+2 = 7 Punkte)

Die folgenden Anforderungen zeigen einen Ausschnitt aus dem ersten Inkrement der Anforderungen eines Aktienverwaltungssystems einer Bank.

A1.1: Nach dem Start der Software muss das System das Startmenü der Software anzeigen.

A2.1: Im Startmenü muss das System der Nutzer*in die Möglichkeit bieten, die Aktienverwaltung auszuwählen.

A2.2: Nachdem die Nutzer*in die Aktienverwaltung gewählt hat, muss das System in den Aktienverwaltungsdialog wechseln.

A2.3: Im Aktienverwaltungsdialog muss das System der Nutzer*in die Möglichkeit bieten, Daten für neue Aktien (AktienID, Aktienname, Einkaufspreis (in Cent als ganze Zahl), Verkaufspreis (in Cent als ganze Zahl)) einzugeben.

A2.4: Nach der vollständigen Eingabe aller Daten einer neuen Aktie muss das System die eingegebenen Daten abspeichern.

A2.5: Im Aktienverwaltungsdialog muss das System der Nutzer*in die Möglichkeit bieten, eine AktienID zur Suche nach dieser Aktie einzugeben.

A2.6: Nach der Eingabe einer AktienID zur Suche muss das System prüfen, ob eine Aktie zur eingegebenen AktienID vorhanden ist.

A2.7: Ergibt die Suche nach einer AktienID, dass diese nicht vorhanden ist, muss das System den Nutzer über die nicht gefundene Aktie informieren.

A2.8: Ergibt die Suche nach einer AktienID, dass diese vorhanden ist, muss das System alle Informationen zu dieser Aktie anzeigen.

A2.9: Für vollständig angezeigte Aktien muss das System der Nutzer*in die Möglichkeit bieten, den Einkaufs- und Verkaufspreis zu ändern.

A2.10: Nach dem Ändern des Einkaufs- und Verkaufspreises muss das System die Aktualisierungen abspeichern.

A3.1: Im Startmenü muss das System der Nutzer*in die Möglichkeit bieten, die Depotverwaltung auszuwählen.

A3.2: Nachdem die Nutzer*in die Depotverwaltung gewählt hat, muss das System in den Depotverwaltungsdialog wechseln.

A3.3: Im Depotverwaltungsdialog muss das System der Nutzer*in die Möglichkeit bieten, Daten für neue Depots (DepotID, Kundennummer; Barreserve (in Cent als ganze Zahl)) einzugeben.

A3.4: Nach der vollständigen Eingabe aller Daten eines neuen Depots muss das System die eingegebenen Daten abspeichern.

A3.5: Im Depotverwaltungsdialog muss das System der Nutzer*in die Möglichkeit bieten, eine DepotID zur Suche nach dem zugehörigen Depot einzugeben.

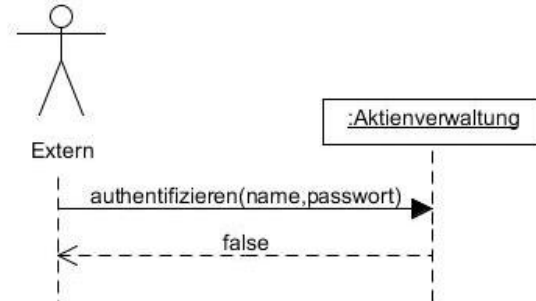
A3.6: Nach der Eingabe einer DepotID zur Suche muss das System prüfen, ob ein Depot zur eingegebenen DepotID vorhanden ist.



- A3.7: Ergibt die Suche nach einer DepotID, dass diese nicht vorhanden ist, muss das System den Nutzer über das nicht gefundene Depot informieren.
- A3.8: Ergibt die Suche nach einer DepotID, dass diese vorhanden ist, muss das System alle Informationen zu diesem Depot anzeigen.
- A3.9: Für vollständig angezeigte Depots muss das System der Nutzer*in die Möglichkeit bieten, einen neuen Depotposten (AktienID, Anzahl der Aktien) einzugeben.
- A3.10: Nach der Eingabe einer AktienID für einen neuen Depotposten muss das System prüfen, ob eine Aktie zur eingegebenen AktienID vorhanden ist.
- A3.11: Ergibt die Suche nach einer AktienID für einen neuen Depotposten, dass diese nicht vorhanden ist, muss das System den Nutzer über die nicht gefundene Aktie informieren.
- A3.12: Ergibt die Suche nach einer AktienID für einen neuen Depotposten, dass diese vorhanden ist, muss das System prüfen, ob die Barreserve ausreicht, den neuen Depotposten zu bezahlen ($\text{Barreserve} - \text{Anzahl der Aktien} * \text{Einkaufspreis} > 0$).
- A3.13: Ergibt die Prüfung der Barreserve beim Anlegen eines neuen Depotpostens, dass die Barreserve nicht ausreicht, muss das System den Nutzer über die nicht mögliche Finanzierung des Kaufs informieren.
- A3.14: Ergibt die Prüfung der Barreserve beim Anlegen eines neuen Depotpostens, dass die Barreserve ausreicht, muss das System den neu angelegten Depotposten zusammen mit einer eindeutigen Depotpostennummer und dem Erstellungsdatum des Depotpostens zum Depot hinzufügen und abspeichern.
- A3.15: Nach dem Speichern eines neuen Depotpostens muss das System die Barreserve des Depots um den Betrag $\text{Anzahl der Aktien} * \text{Einkaufspreis}$ reduzieren.
- A4.1: Für vollständig angezeigte Depots muss das System der Nutzer*in die Möglichkeit bieten, eine Depotpostennummer zur Suche nach dem zugehörigen Depotposten einzugeben.
- A4.2: Ergibt die Suche nach einer Depotpostennummer, dass diese nicht vorhanden ist, muss das System den Nutzer über den nicht gefundenen Depotposten informieren.
- A4.3: Ergibt die Suche nach einer Depotpostennummer, dass diese vorhanden ist, muss das System alle Informationen zu diesem Depotposten anzeigen.
- A4.4: Für vollständig angezeigte Depotposten muss das System der Nutzer*in die Möglichkeit bieten, die Anzahl der Aktien zu ändern.
- A4.5: Nach der Erhöhung der Anzahl der Aktien eines Depotpostens muss das System prüfen, ob die Barreserve ausreicht, die Erhöhung der Aktienzahl zu bezahlen ($\text{Barreserve} - (\text{neue Anzahl der Aktien} - \text{alte Anzahl der Aktien}) * \text{Einkaufspreis} > 0$).
- A4.6: Ergibt die Prüfung der Barreserve beim Erhöhen der Anzahl der Aktien eines Depotpostens, dass die Barreserve nicht ausreicht, muss das System den Nutzer über die nicht mögliche Finanzierung des Kaufs informieren.
- A4.7: Ergibt die Prüfung der Barreserve beim Erhöhen der Anzahl der Aktien eines Depotpostens, dass die Barreserve ausreicht, muss das System den Anzahlwert des Depotpostens ändern und abspeichern.
- A4.8: Nach dem Speichern Erhöhung der Anzahl der Aktien eines Depotpostens muss das System die Barreserve des Depotpostens um den Betrag $(\text{neue Anzahl der Aktien} - \text{alte Anzahl der Aktien}) * \text{Einkaufspreis}$ verringern.
- A4.9: Bei der Verringerung der Anzahl der Aktien eines Depotpostens muss das System die Barreserve des Depotpostens um den Betrag $(\text{alte Anzahl der Aktien} - \text{neue Anzahl der Aktien}) * \text{Verkaufspreis}$ erhöhen.
- G1.1: In allen Teildialogen (Aktienverwaltungsdialog, Depotverwaltungsdialog, ...) muss das System der Nutzer*in die Möglichkeit bieten, zurück zum Startmenü der Software zu kehren.
- G1.2: Nach Auswahl der Rückkehrmöglichkeit zum Startmenü in einem Teildialog muss das System das Startmenü der Software anzeigen.
- a) Leiten Sie aus den Anforderungen schrittweise ein Klassendiagramm ab. Beachten Sie, dass bereits bei der Erstellung des Analysemodells möglichst darauf geachtet werden soll, dass Funktionalität nicht doppelt implementiert wird. Auf Elemente der Oberfläche kann bei der Modellierung verzichtet werden, da die Oberfläche häufig getrennt entwickelt wird. Typischerweise finden Sie Verwaltungsklassen und zugehörige Entitätsklassen, wobei Entitätsklassen üblicherweise nicht auf Verwaltungsklassen zugreifen.
- b) Geben Sie jeweils neue Sequenzdiagramme für die Abläufe
- „neue Aktie erstellen“,
 - „Aktieneinkaufspreis ändern“,
 - „neues Depot anlegen“,
 - „Depot erfolgreich suchen, dann im Depot Depotposten erfolglos suchen“,
 - „Aktie erfolgreich zum Depot hinzufügen mit Parametern (depotId, aktienId, anzahl)“,



„Aktienanzahl eines Depotpostens depotposId im Depot depotId erfolglos erhöhen“
an. Beim Erstellen der Sequenzdiagramme können Sie davon ausgehen, dass die ersten Aufrufe der Funktionalität durch einen abstrakten externen Aktor „Extern“ durchgeführt werden. Dieser Aktor kann für das GUI oder weitere Klassen stehen. Soll ein Objekt dann eine Meldung an den Nutzer ausgeben, kann diese einfach an den externen Aktor geschickt werden. Hätten Sie z. B. eine Klasse Aktienverwaltung, bei der man sich zunächst authentifizieren muss (was nicht gefordert ist), hat die Klasse Aktienverwaltung eine Methode `authentifizieren(name,password)`. Ein Diagramm für eine nicht erfolgreiche Authentifizierung wird dann wie rechts gezeigt aussehen. Eventuell müssen Sie weitere Methoden ergänzen, die sich indirekt aus den Anforderungen ergeben. Dokumentieren Sie Änderungen, die sie in Ihrem Klassendiagramm auf Grundlage der Sequenzdiagramme vorgenommen haben.



Hinweis: ausgefüllte schwarze Pfeile haben in UMLet die Form `lt=<<<-` (Sie können die Sequenzdiagramme auch von Hand zeichnen.)