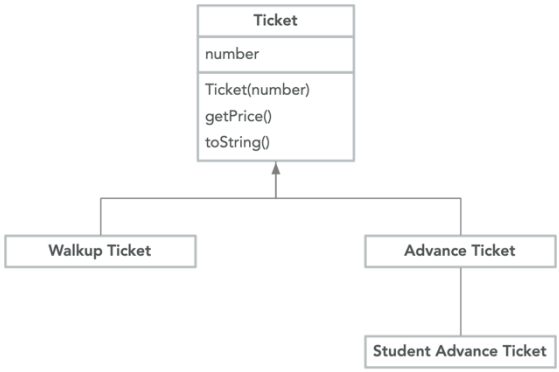
BEISPIELSAMMLUNG

Objektorientierte Programmierung

Table of Contents

Ticket for events

Consider the task of representing types of tickets to events. Each ticket has a unique number and a price. There are three types of tickets: walk-up tickets, advance tickets, and student advance tickets:



- Walk-up ticket are purchased the day of the event and cost $50
- Advance tickets purchased 10 or more days before the event cost $30, and advance tickets purchased fewer than 10 days before the event cost $40.
- Student advance tickets are sold at half the price of normal advance tickets: when they are purchased 10 or more days early they cost $15, and when they are purchased fewer than 10 days early they cost $20.

1. Create a project called TicketMain.
2. Implement a class called Ticket that will serve as the superclass for all three types of tickets. Define all common operations in this class, and specify all differing operations in such a way that every subclass must implement them. Define the following operations:
   - The ability to construct a ticket by number.
   - The ability to ask for a ticket's price
   - The ability to prinln a ticket object as a String. An example String would be "Number: 17, Price $50.00"
     Create a ticket object and call each method.
3. Implement a class called WalkupTicket to represent a walk-up event ticket. Walk-up tickets are also constructed by number, and they have a a price of $50. Create a walk-up ticket object and call each method.
4. Implement a class called AdvanceTicket to represent tickets purchased in advance. An advance ticket is constructed with a ticket number and with the number of days in advance that the ticket was purchased. Advance tickets purchased 10 or more days before the event and cost $40. Create an advance ticket object and call each method.

5. Implement a class called StudentAdvanceTicket to represent tickets purchased in advance by students. A student advance ticket is constructed with a ticket number and with the number of days in advance that the ticket was purchased. Student advance tickets purchased 10 or more days early they cost $15, and when they are purchased frewer than 10 days early they cost $20. When a student advance ticket is printed, the String should mention that the student must show his or her ID (for example, "Number: 17, Price $15.00 (ID required)". Create a student advance ticket object and call each method.

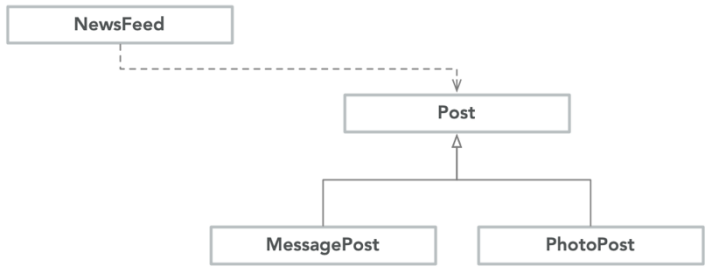
Person, Schüler und Lehrer

1. Erstelle ein Projekt mit dem Namen SchulApp.
2. Schreibe eine Klasse Person.
   Eine Person hat einen Vornamen und einen Nachnamen. Beachte das Geheimnisprinzip.
   Schreibe einen Konstruktor.
   Schreibe get- und set-Methoden zum Abfragen und Ändern der beiden Namen. Schreibe eine Methode toString, die den Namen der Person zurückgibt. Erzeuge ein Person-Objekt, ändere den Namen und gib die Person aus.
3. Schreibe eine Klasse Schueler, die von Person erbt.
   Ein Schüler hat zusätzlich eine Katalognummer. Beachte das Geheimnisprinzip.
   Schreibe einen Konstruktor. Rufe den Konstruktor der Oberklasse auf.
   Schreibe get- und set-Methoden zum Abfragen und Ändern der Katalognummer. Überschreibe die Methode toString. Rufe die Methode der Oberklasse auf. Erzeuge ein Schüler-Objekt, ändere die Katalognummer und gib den Schüler aus.
4. Schreibe eine Klasse Lehrer, die von Person erbt.
   Ein Lehrer hat zusätzlich eine id, die aus den ersten 3 Buchstaben des Nachnamens und aus dem Anfangsbuchstaben des Vornamens (z.B. SilJ) besteht.
   Schreibe einen Konstruktor. Rufe den Konstruktor der Oberklasse auf.
   Schreibe eine get-Methode zum Abfragen der id.
   Überschreibe die Methode toString. Rufe die Methode der Oberklasse auf. Erzeuge ein Lehrer-Objekt und gib den Lehrer aus.

Newsfeed für ein soziales Netz

Diese Anwendung implementiert den Newsfeed-Teil für ein soziales Netz. Der Newsfeed ist die Liste der Nachrichten, die angezeigt wird, wenn ein Benutzer die Hauptseite des Netzwerks öffnet.

Im folgenden Klassendiagramm sieht man, dass der Newsfeed Textnachrichten und Fotoeinsendungen unterstützen soll:



1. Erstelle ein Projekt mit dem Namen NewsfeedMain.
2. Schreibe eine Klasse Post.
   Die Klasse Post dient als Oberklasse für die spezielleren Klassen MessagePost und PhotoPost.
   Jedes Post-Objekt speichert den Benutzernamen des Einsenders (username). Schreibe einen Konstruktor mit einem Parameter username.
   Schreibe eine Methode display, die die Details des Posts anzeigt.
   Erzeuge ein Post-Objekt.
3. Schreibe eine Klasse MessagePost, die von Post erbt.
   Die Klasse MessagePost ist für Textnachrichten zuständig.
   Jedes MessagePost-Objekt speichert zusätzlich zum Benutzernamen die Textnachricht (message).
   Schreibe einen Konstruktor mit den Parametern username und message.
   Schreibe eine Methode getMessage.

Überschreibe die Methode display, die die Details der Textnachricht anzeigt.
Erzeuge ein MessagePost-Objekt.

4. Schreibe eine Klasse PhotoPost, die von Post erbt.
   Die Klasse PhotoPost ist für Fotoeinsendungen zuständig.
   Jedes PhotoPost-Objekt speichert zusätzlich zum Benutzernamen den Namen der Bilddatei (filename) und die Bildbeschriftung (caption).
   Schreibe einen Konstruktor mit den Parametern username, filename und caption. Schreibe die Methoden getFilename und getCaption.
   Überschreibe die Methode display, die die Details der Fotoeinsendung anzeigt. Erzeuge ein PhotoPost-Objekt.
5. Schreibe eine Klasse Newsfeed.
   Die Klasse Newsfeed verwaltet die Posts in einer ArrayList (posts).
   Schreibe eine Methode addPost mit einem Parameter post, die post in die ArrayList aufnimmt.
   Schreibe eine Methode show, die die Details aller Posts anzeigt.
   Füge dem Newsfeed mindestens eine Textnachricht und eine Fotoeinsendung hinzu.
6. Erweitere die Klasse Post um Likes.
   Definiere ein Feld likes, das zählt, wie vielen Lesern die Nachricht gefällt. Schreibe eine Methode like, die likes erhöht und eine Methode unlike, die likes vermindert (like darf nicht negativ werden).
   Gib die Anzahl der likes in display aus.

7. Erweitere die Klasse Post um einen Zeitstempel.
   Speichere für jede Nachricht den Zeitstempel (timestamp) als long.
   Beschaffe dir im Konstruktor vom Java-System mit System.currentTimeMillis die Systemzeit als long-Wert in Millisekunden.
   Erweitere die Methode display so, dass z. B. vor 5 Minuten angezeigt wird. Benutze dazu eine Hilfsmethode timeString, die den in der Vergangenheit liegenden übergebenen Zeitstempel im Vergleich zur aktuellen Zeit als String beschreibt.
8. Erweitere die Klasse Post um Kommentare. Speichere die Kommentare in einer ArrayList. Schreibe eine Methode addComment.
   Gib die Anzahl der Kommentare in display aus.

Medien-Datenbank

Schreibe ein Java-Programm, das Informationen über Medien wie Musikalben, Filme, etc. verwaltet. Folgende Details eines Musikalbums wollen wir erfassen: Titel des Albums, Künstler (Name der Band oder der Sängerin), Kommentar (ein beliebiger Text).

Folgende Details sollen für einen Film erfasst werden: Titel des Films, Regisseur (Name des Regisseurs), Kommentar (ein beliebiger Text).

1. Schreibe eine Klasse Medium. Die Klasse Medium dient als Oberklasse für die spezielleren Medientypen Musikalbum und Film.
   Schreibe einen Konstruktor, der den Titel setzt.
   Schreibe get- und set-Methoden für den Titel und den Kommentar.
   Schreibe eine Methode toString, die das Medium als String zurückgibt, z.B. „Titel: 25, Kommentar: ...“
   Teste die Klasse Medium.
2. Schreibe eine Klasse Musikalbum, die von Medium erbt. Diese Klasse verwaltet Informationen über Musikalben.
   Schreibe einen Konstruktor, der den Titel und den Künstler setzt.
   Schreibe get- und set-Methoden und eine toString-Methode.
   Teste die Klasse Musikalbum.
3. Schreibe eine Klasse Film, die von Medium erbt. Diese Klasse verwaltet Informationen über Filme.
   Schreibe einen Konstruktor, der den Titel und den Regisseur setzt.
   Schreibe get- und set-Methoden und eine toString-Methode.
   Teste die Klasse Film.
4. Schreibe eine Klasse Datenbank.
   Diese Klasse verwaltet die einzelnen Musikalben und Filme in einer ArrayList mit Medium-Objekten.
   Schreibe eine Methode erfasseMedium, die ein neues Medium in die ArrayList aufnimmt.
   Schreibe eine Methode listeAuf, die alle Medien auf der Konsole anzeigt.
   Tipp: Rufe toString von Musikalbum bzw. Film auf.
   Teste die Klasse Datenbank.

Tickets of events (improved)

Implement a class called Ticket that will serve as the superclass for all three types of tickets. Define all common operations in this class, and specify all differing operations in such a way that every subclass must implement them.

No actual objects of type Ticket will be created: Each actual ticket will be an object of a subclass type.

Newsfeed für soziales Netz (verbessert)

Verhindere das Erzeugen von Post-Objekten. Man soll nur MessagePost- oder PhotoPost- Objekte erzeugen können.

Wie man etwas isst

1. Schreibe ein Interface Essbar.
   Schreibe eine abstrakte Methode wieManEsIsst, die einen String zurückgibt.
2. Schreibe eine Klasse Tier.
   Deklariere eine Eigenschaft wie z. B. gewicht und schreibe einen Konstruktor.
3. Schreibe eine Klasse Huhn, die von Tier erbt und das Interface Essbar implementiert.
   Implementiere die wieManEsIsst-Methode, z. B. Huhn: Brate es.
   Test die Klasse Huhn. Rufe die wieManEsIsst-Methode auf.
4. Schreibe eine Klasse Obst.
   Deklariere eine Eigenschaft wie z. B. farbe und schreibe einen Konstruktor.
5. Schreibe eine Klasse Apfel, die von Obst erbt und das Interface Essbar implementiert. Implementiere die wieManEsIsst- z.B. Apfel: Mach' Apfelsaft draus.
   Test die Klasse Apfel. Rufe die wieManEsIsst-Methode auf.