

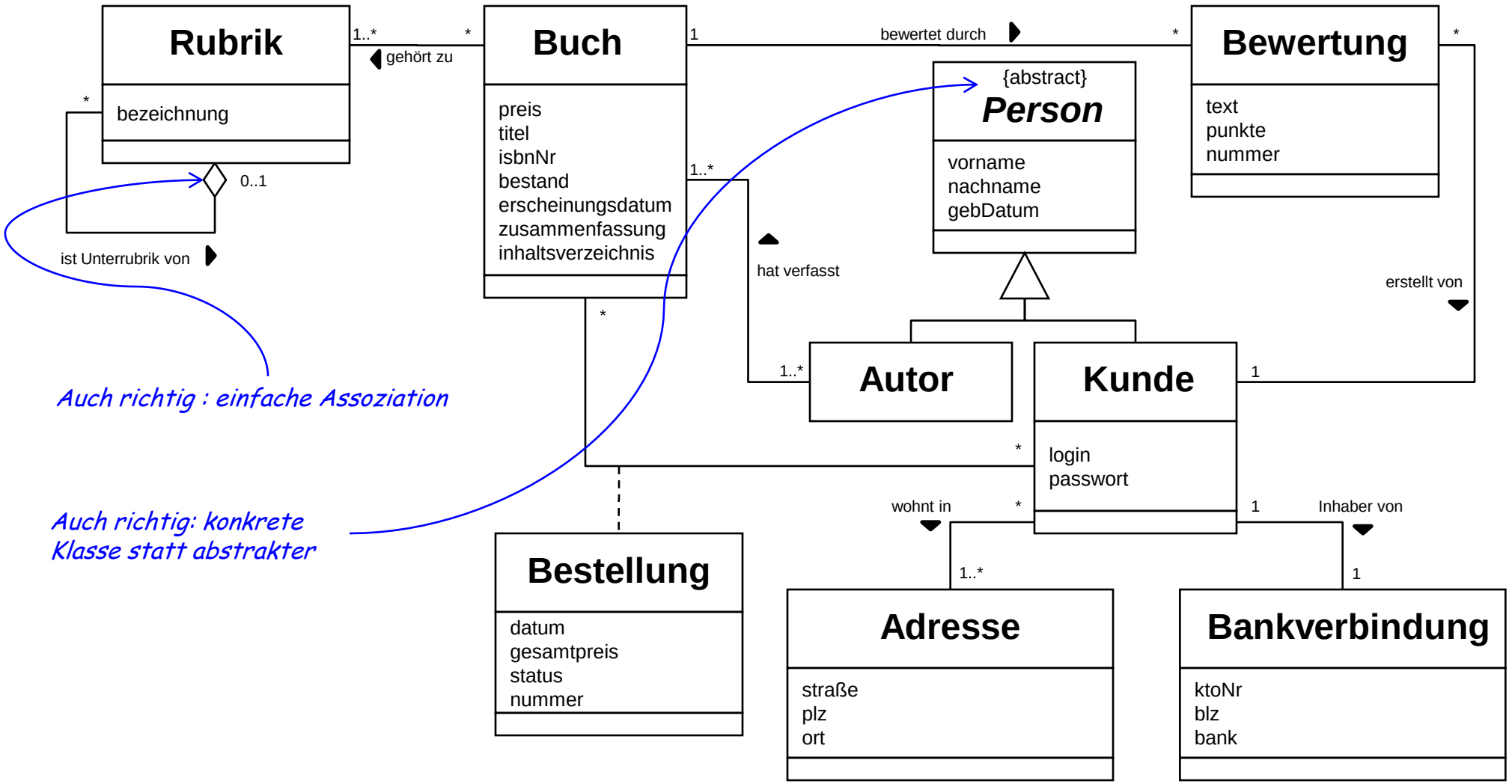
# Übung
# „Einführung in die Softwaretechnik“

*Lösungshinweise zum Übungsblatt 10*

# Aufgabe 22

1. Gründe für Vernachlässigung des Requirements Engineering in der Praxis
   - *Allgemein*:
     - Mangelnde RE-Kompetenz (auf beiden Seiten)
       (Wichtigkeit/Priorität der Anforderungsspezifikation nicht immer geklärt,
       Je nach Projektgröße erscheint ausführliche RE-Phase unnötig bzw. zu teuer/lang)
     - Administrative Gründe
       (Mitarbeiter haben zu wenig projektbezogene Zeit, Budgets)
     - Kosten und Nutzen des RE nicht immer kommunizierbar
       (Produkt ist Code, Nutzen der RE-Ergebnisse für Kunden)
   - *Einzellösungen* für Kunden
     - Auf Kundenseite mangelnde Kompetenz/Frequenz/Priorität der Kommunikation bzgl. Anforderungen (z.B. Mitarbeiter-Freistellung)
     - Bezahlung für RE-Phase durch Kunden nicht immer möglich/gewünscht
   - *Massenlösungen* für Markt
     - Kein spezifischer Kunde vorhanden (Finanzierung von RE)
     - Zeitdruck für Markteinführung des Endprodukts vorrangig („Time-to-Market“)
2. Situationen, in denen das Ausführen von Arbeiten anderer Phasen beim RE erforderlich wird:
   - Prototyp-Erstellung
   - Machbarkeitstests, Vorstudien (z.B. Schnittstellen-Exploration bestehender Systeme bei Ablösung)

# Aufgabe 23a mit Assoziationsklasse

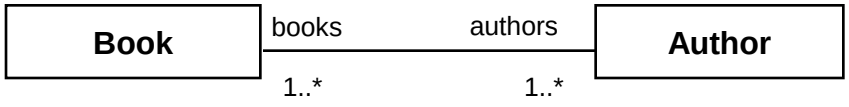


# Aufgabe 23b

Book.java

```
import java.util.*;
public class Book {
    private Collection<Author> authors = new HashSet<Author>();
    public Collection<Author> getAuthors() {
        return authors;
    }
    public void addAuthor(Author a) {
        if (a != null) {
            if (!this.authors.contains(a)) {
                this.authors.add(a);
                a.addBook(this);
            }
        }
    }
    public void removeAuthor(Author a) {
        if (a != null) {
            if (this.authors.contains(a)) {
                this.authors.remove(a);
                a.removeBook(this);
            }
        }
    } …
}
```

*Endlosschleifen beheben*

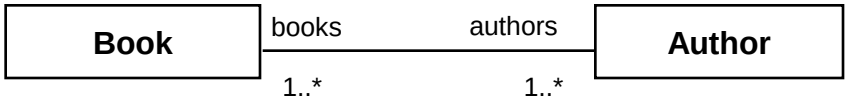


# Aufgabe 23b

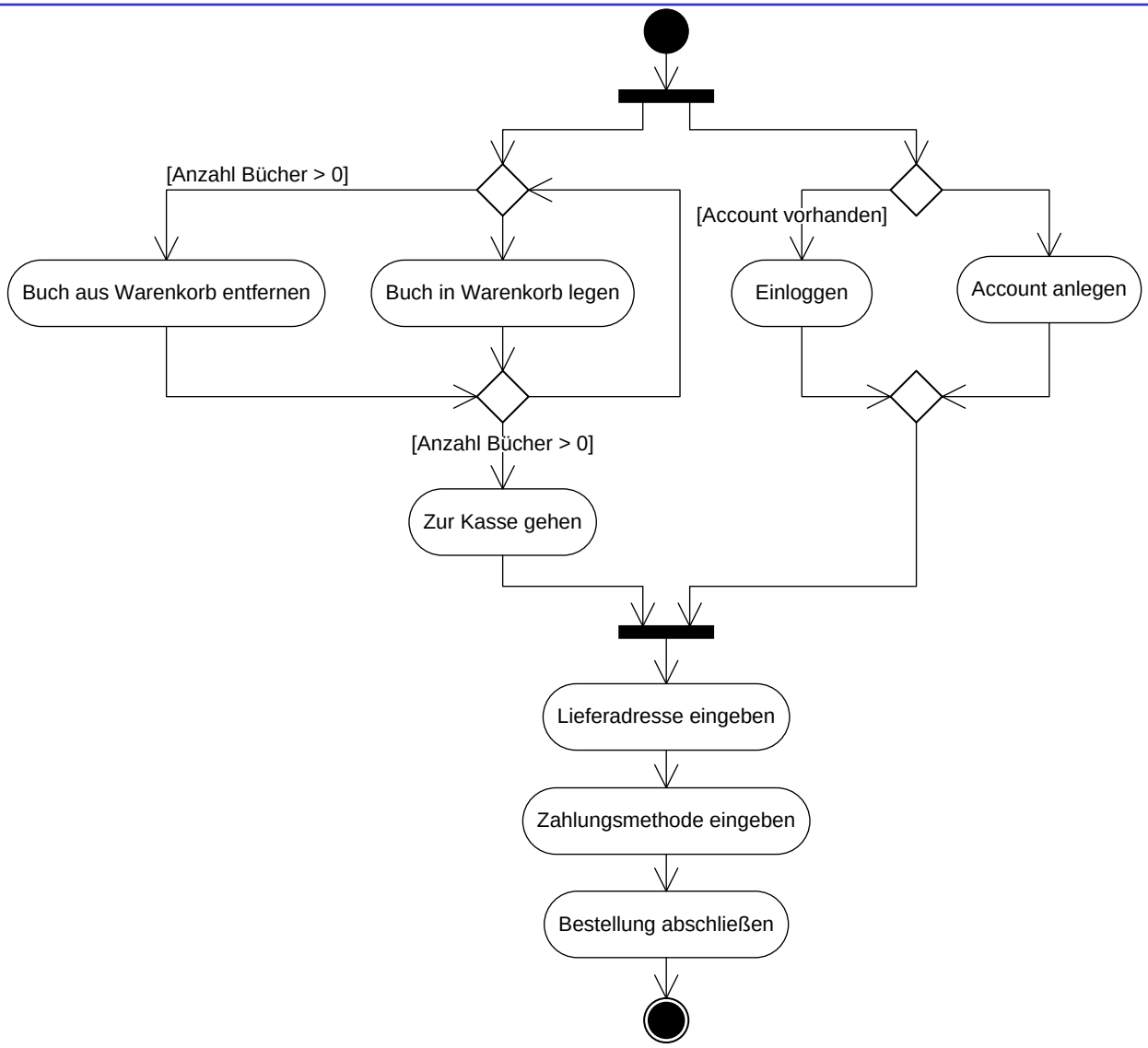
Author.java

```
import java.util.*;
public class Author {
    private Collection<Book> books = new HashSet<Book>();
    public Collection<Book> getBooks() {
        return books;
    }
    public void addBook(Book b) {
        if (b != null) {
            if (!this.books.contains(b)) {
                this.books.add(b);
                b.addAuthor(this);
            }
        }
    }
    public void removeBook(Book b) {
        if (b != null) {
            if (this.books.contains(b)) {
                this.books.remove(b);
                b.removeAuthor(this);
            }
        }
    } ...
}
```

*Nullpointer vermeiden*



# Aufgabe 24


[Anzahl Bücher > 0]
[Account vorhanden]
Buch aus Warenkorb entfernen
Buch in Warenkorb legen
Einloggen
Account anlegen
[Anzahl Bücher > 0]
Zur Kasse gehen
Lieferadresse eingeben
Zahlungsmethode eingeben
Bestellung abschließen