Kapitel 12 – Fehlerbehandlung: Zusammenfassung + Übungen

In diesem Kapitel lernst du, wie man in Java Fehler erkennt und behandelt. Mit try-catch-Blöcken kannst du Programmabbrüche vermeiden und gezielt auf Ausnahmen reagieren.

Themenübersicht

1. Ausnahme (Exception)

Ein Fehler, der während der Programmausführung auftritt, z. B. Division durch 0 oder Zugriff auf null.

2. try-catch-Block

Code, der Fehler verursachen könnte, wird in `try` geschrieben. Fehler werden in `catch` behandelt.

3. finally-Block

Wird immer ausgeführt, egal ob ein Fehler auftritt oder nicht. Ideal zum Aufräumen.

4. throw

Mit `throw` kann man selbst eine Exception auslösen. z. B.: `throw new IllegalArgumentException();`

5. throws

Gibt an, dass eine Methode eine Exception weiterreichen kann: `public void liesDatei() throws IOException`

6. Checked vs. Unchecked Exceptions

- Checked: Müssen behandelt oder weitergegeben werden (z. B. IOException)
- Unchecked: Müssen nicht behandelt werden (z. B. NullPointerException)

7. Mehrere catch-Blöcke

Verschiedene Fehlertypen können separat behandelt werden.

Übungsaufgaben

1. 1. Schreibe ein Programm, das eine Division durch null versucht und den Fehler auffängt.

2. 2. Baue einen try-catch-Block, der eine `NumberFormatException` bei fehlerhafter Zahleneingabe behandelt.
3. 3. Ergänze den Block mit `finally`, der 'Fertig!' ausgibt – egal ob ein Fehler kam oder nicht.
4. 4. Schreibe eine Methode `wurzel(double x)`, die eine `IllegalArgumentException` wirft, wenn x < 0 ist.
5. 5. Verwende `throws`, um in einer Methode eine `IOException` an den Aufrufer weiterzugeben.
6. 6. Erkläre den Unterschied zwischen checked und unchecked Exceptions anhand eines Beispiels.
7. 7. Baue ein Beispiel mit zwei catch-Blöcken für unterschiedliche Exceptions.
8. 8. Was passiert, wenn man eine checked Exception weder behandelt noch weitergibt?