PROGRAMMIER-
TECHNIK I
ÜBUNG

13

Die Besprechung des Aufgabenblattes erfolgt am 18/19 Januar 2021. Das zweite Wiederholungsblatt stelle ich zur nächsten Übung zur Verfügung.

Aufgabe 1 (Exception Handling, eigene Exceptions definieren)

- Schritt 1: Erzeugen Sie eine Klasse „`PruefungsPanik`“ die von `Exception` erbt.
- Schritt 2: Definieren Sie einen parameterlosen Konstruktor für die Klasse „`PruefungsPanik`“, der nichts Anderes macht, als den `super`-Konstruktor aufzurufen.
- Schritt 3: Implementieren Sie einen Kontruktor mit einem String als Parameter, der den `super-Konstruktor` mit eben diesem Parameter aufruft.
- Schritt 4: Implementieren Sie eine Klasse `TurboLernenFuerPruefungen`. Schreiben Sie hier eine Methode „`lernen`“, die potenziell die Exception „`PruefungsPanik`“ auslöst:

  ```
  public void lernen (boolean nochGenugZeit)

        ist nochGenugZeit true → „Großartiges Zeitmanagement. Weiter so!“ ausgeben.

        ist nochGenugZeit false → Werfen Sie die Exception „PruefungsPanik“ aus mit
        der Meldung „In der Ruhe liegt die Kraft.“.
  ```

- Schritt 5: Implementieren Sie eine Klasse `TurboLernenFuerPruefungenTestlauf`.

  Testen Sie Ihre Klasse `TurboLernenFuerPruefungen` indem Sie die Methode „`lernen`“ aufrufen. Kann man die Methode „lernen“ ohne try/catch-Block aufrufen? Warum (nicht)?

  Behandeln Sie die mögliche `Exception`, die beim Aufruf der Methode „`lernen`“ geworfen werden kann. Testen Sie nun den Methodenaufruf mit lernen (true) und lernen (false).

Aufgabe 2 (Polymorphe Exceptions)

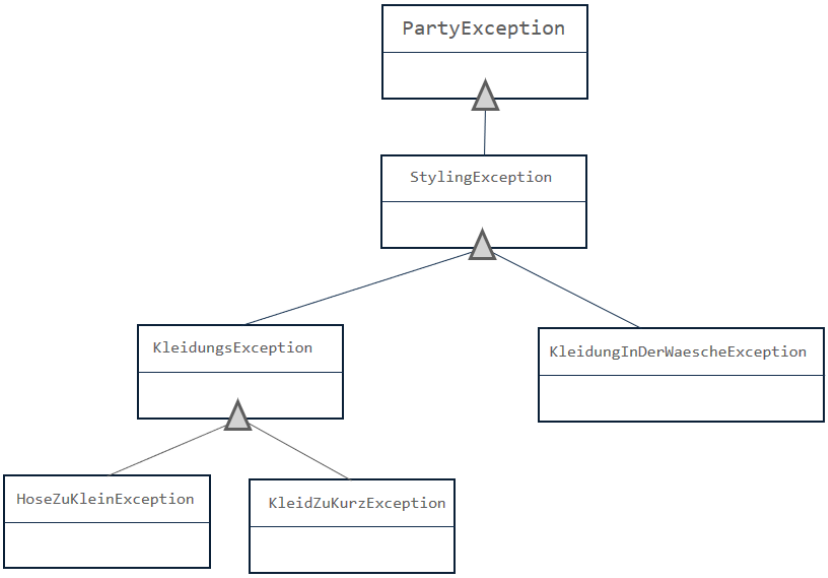
Nehmen Sie an, der Code in dem nachfolgenden try-catch-Block ist zulässig. Zeichnen Sie zwei verschiedene Beispiele für Klassendiagramme, die die Exception-Klassen exakt repräsentieren, so dass der try-catch-Block zulässig ist. Fangen Sie die Darstellung der Klassenhierarchie bei der Exception-Klasse selbst an. Attribute und Methoden dürfen Sie vernachlässigen.

```
try {
	ich.geheRisikoEin();
catch (MeineException m) {
	// Wiederherstellen nach Exception m
}
catch (DeineException d) {
	// Wiederherstellen nach Exception d
}
catch (IhreException i) {
	// Wiederherstellen nach Exception i
}
catch (SeineException s) {
	// Wiederherstellen nach Exception s
}
```

Aufgabe 3 (try-catch-Reihenfolge und Vererbungshierarchie bei Exceptions)

Gegeben sei das nachfolgende Klassendiagramm. PartyException erbt von Exception (nicht in der Grafik dargestellt). Versuchen Sie, die Aufgabe zuerst ohne Rechner zu lösen. Wenn Sie fertig sind, überprüfen Sie Ihre Ergebnisse.

1. Schreiben Sie die Klassendefinition für die abgebildete Exception-Hierarchie.
2. Schreiben Sie zwei zulässige try-catch-Blöcke, die das abgebildete Klassendiagramm exakt repräsentieren (polymorphe Ausnahmebehandlung möglich). Gehen Sie von der Annahme aus, dass alle Exceptions von der Methode innerhalb des try-catch-Blocks ausgelöst werden können.




```
Syntax für einen try/catch-Aufruf

try {
	// riskanter Code, den wir hier nicht näher spezifizieren
} catch (MeineException e) {
	// Code, der das Wiederherstellen der nach der Exception übernimmt
}
```

Aufgabe 4 (Exceptions im Konstruktor definieren)

Auf dem Aufgabenblatt 10 haben Sie eine Klasse „Hund“ definiert und im Konstruktor sichergestellt, dass das Alter >= 0 ist. Definieren Sie eine eigene Exception für diesen Fall und erweitern Sie den Konstruktor entsprechend. Testen Sie den Konstruktor bezüglich des Exception Handlings.

Hier nochmals die „alte“ Aufgabe:

Aufgabe Schreiben Sie eine Klasse Hund, die einen Hund mit Namen und Alter repräsentieren soll.

- Definieren Sie einen Konstruktor, der die Instanzvariablen initialisiert. Stellen Sie sicher, dass das Alter >= 0 ist.
- Schreiben Sie Getter und Setter für beide Instanzvariablen. Stellen Sie auch hierbei sicher, dass das Alter immer >= 0 ist. Geben Sie eine entsprechende Fehlermeldung aus, wenn dies nicht der Fall ist.
- Schreiben Sie eine Methode alterInPersonenJahren, die das Hundealter zurückgibt. Das Alter erhalten Sie wenn das Hundealter mit 7 multipliziert wird.
- Schreiben Sie eine Methode toString, die eine Zeichenkettenrepräsentation von Hundobjekten (z.B. Name: MeinHund; Alter: 3 (in Menschenjahren: 21))
- Schreiben Sie eine Testklasse in der Sie Ihre Methoden testen