HFT Stuttgart
Schellingstraße 24
70174 Stuttgart

Programmieren 1

9 Übung

9.1 Wurzeln ziehen (leicht)

Implementieren Sie die statische Methode **`public static double zieheWurzel(double x)`**. Diese Methode soll die Wurzel von x ziehen. Zur Berechnung können Sie die von Java gelieferte Methode `Math.sqrt()` nutzen, welche für negative Zahlen das Ergebnis NaN liefert.

Ihre Methode soll nun eine Exception vom Typ **`MatheRuntimeException`** werfen, wenn Sie von einer negativen Zahl die Wurzel ziehen. Erstellen Sie die entsprechende **`MatheRuntimeException`**-Klasse. Rufen Sie dann die Methode **`zieheWurzel()`** aus **`main()`** mit **`try/catch`** auf und probieren Sie verschiedene positive und negative Argumente.

9.2 MatheBibliothek (mittel)

Gegeben ist die folgende (noch unvollständige) Klasse `MatheBibliothek`:

```
public class MatheBibliothek {

	public int berechne(int n) {

		if(n == 5)

			throw new EineAusnahme(n);

		if(n == 10)

			throw new EineZweiteAusnahme(n);

		if(n < 0)

			throw new EineRuntimeAusnahme(n);

		if(n == 0)

			throw new NullPointerException();

		return n * 2;

	}

}
```

Erstellen Sie zunächst die noch fehlenden Exceptions: `EineAusnahme` und `EineZweiteAusnahme` sollen `Exceptions` sein, `EineRuntimeAusnahme` soll eine `RuntimeException` sein. Definieren Sie den Konstruktor mit Hilfe des Aufrufs super(String message) so, dass in der Fehlermeldung die übergebene Zahl n ausgegeben wird. Ergänzen Sie den Kopf der Methode `berechne` um die fehlenden throw-Anweisungen – der Rumpf der Methode soll nicht verändert werden.

Realisieren Sie dann die folgende Klasse:

```
public class Main {

	public static void test(MatheBibliothek m) {

		for(int i = -10; i <= 10; i++){

				System.out.print("Berechnung für " + i + " ergibt ");

				System.out.println(m.berechne(i));

		}

	}

	public static void main(String[] args) {

		MatheBibliothek m = new MatheBibliothek ();

		test(m);

	}

}
```

Korrigieren/ändern Sie die Methode `test()` folgendermaßen:

- Die Methode `m.berechne(i)` soll für alle 21 Werte von `i` aufgerufen werden.
- `EineAusnahme` soll dabei (mit einer Meldung am Bildschirm) gefangen werden.
- `EineZweiteAusnahme` soll dabei (mit einer Meldung am Bildschirm) gefangen werden. In diesem Fall soll eine (noch zu definierende Exception) `EineDritteAusnahme` geworfen werden. Diese Ausnahme soll nicht weiter behandelt (sondern bis zum Ende weitergereicht) werden.
- Die beiden RuntimeExceptions `EineRuntimeAusnahme` und `NullPointerException` sollen in einem einzigen Catch-Block gefangen werden und ebenfalls eine Meldung am Bildschirm ausgeben.
- Im Finally-Block soll nochmals die Variable i ausgegeben werden.

9.3 Datum (umfangreich)

Ein Datum kann als eine Zeichenkette der Form „tt.mm.jjjj“ dargestellt werden. Ausgehend von solch einer Zeichenkette sollen Sie in dieser Aufgabe ein passendes Datumsobjekt konstruieren. Auf der Moodle-Seite finden Sie eine Implementation der Klasse **`Datum.java`**. Machen Sie sich als erstes mit dieser Klasse vertraut.

Bei der Umwandlung einer Zeichenkette in ein Datumsobjekt können folgende Problemfälle eintreten:

- Die Zeichenkette kann fehlerhaft formatiert sein, d.h. sie ist nicht von obiger Form. Solche Fehler sollen eine Ausnahme vom Typ **`DatumFormatException`** werfen.
- Die Zeichenkette kann ein ungültiges Datum angeben, selbst wenn sie richtig formatiert ist (z.B. 45.13.2021). Solche Fehler sollen eine Ausnahme vom Typ **`DatumUngueltigException`** werfen.

Erstellen Sie eine Klasse **`DatumsHelfer`** mit:

a) Zwei statischen inneren Klassen **`DatumFormatException`** und **`DatumUngueltigException`** mit jeweils zwei Konstruktoren: ein Standardkonstruktor und ein Konstruktor mit dem Argument `message` vom Typ `String`.

b) Schreiben Sie eine statische Methode `Datum parseDatum(String s)`, die eine Zeichenkette interpretiert und in ein Datumsobjekt umwandelt. Dabei sollen die obigen Ausnahmen geworfen werden, wenn die Zeichenkette kein korrektes Datum enthält.
Zum Umwandeln von Zahlen kann die Methode `Integer.parseInt` verwendet werden, die für ungültige Zahlen selbst eine Ausnahme wirft.
Zum Testen, ob ein Datum gültig ist, kann die Methode `Datum.istGueltig` verwendet werden.

c) Übergeben Sie beim Erzeugen der Ausnahmen sinnvolle Fehlermeldungen an den entsprechenden Konstruktor.

d) Schreiben Sie eine statische Methode `testDatum`, die folgende Zeichenketten in ein Datumsobjekt umwandelt und den Folgetag ausgibt. Tritt ein Fehler auf, soll stattdessen die in der Ausnahme gespeicherte Nachricht ausgegeben werden.
   - „14.05.2021“
   - „1.1.2021“
   - „32.05.2021“
   - „14th May“