2. Praktische Leistungsfeststellung

4ACIF

Account/Passwort

ZUNAME Vorname

Datum 2023-06-14

Uhrzeit 18:50 - 20:20

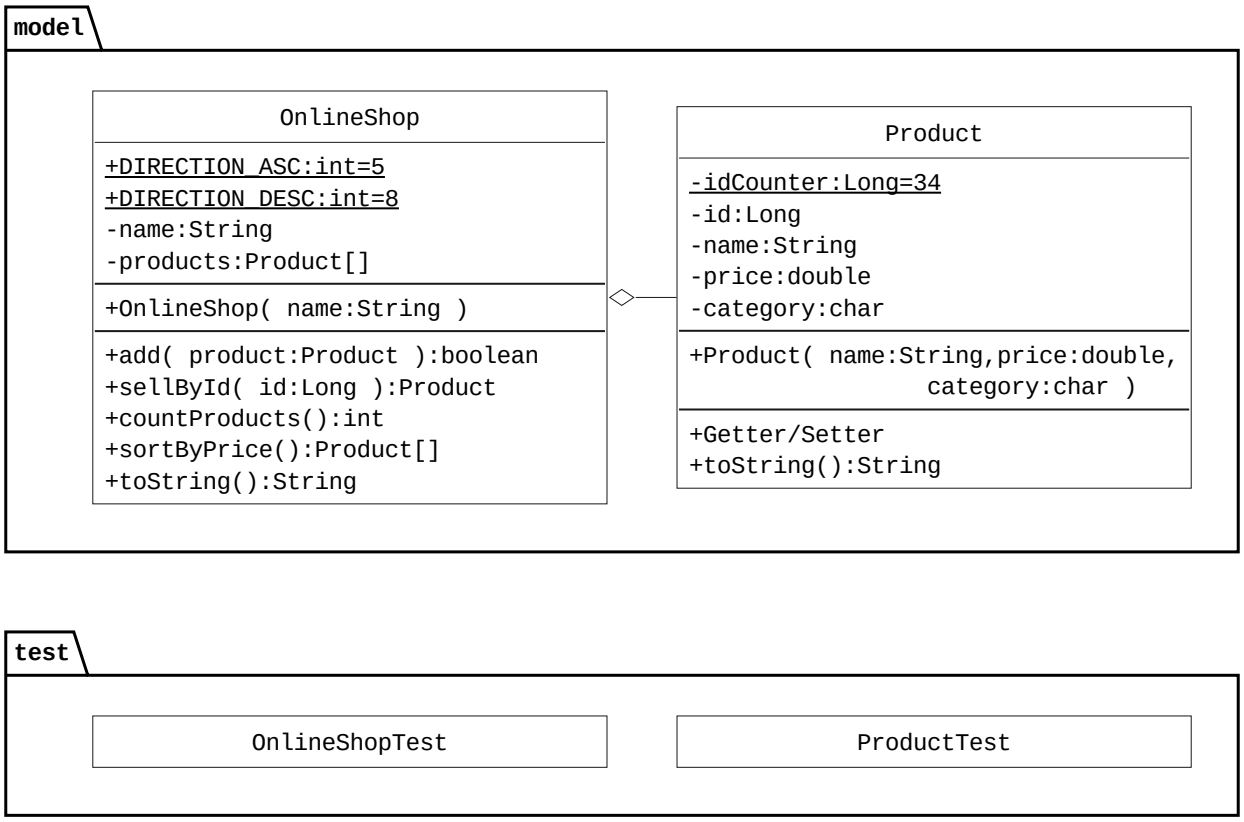
Prüfer: SCHLAG Martin (SLM)

- Bitte achten Sie darauf, nur kompilierfähige Programme abzugeben. Sollten Sie einen Compilerfehler nicht beheben können, kommentieren Sie das entsprechende Codestück aus. Nicht kompilierfähige Programme werden **negativ** bewertet!
- Halten Sie sich **exakt** an die Vorgaben, Formate sowie Klassen- und Methodennamen.

Beschreibung

Die beiden Unternehmer *Ab Zocker* und *Gaune Rei* benötigen für ihr neues Starup-Unternehmen einen Prototyp für einen Onlinehandel, bei dem unterschiedliche Produkte verwaltet und zum Kauf angeboten werden können. Entwickeln Sie die Software auf Basis des UML-Diagramms und den gegebenen Anforderungen.

UML-Diagramm



Prolog

- Angabe auf dem **Z: Laufwerk** entpacken.
- Entpacktes Projekt `s4-plf-001_online_shop` auf den Desktop kopieren.
- **ALLE** Dateien und Verzeichnisse vom **Z: Laufwerk** löschen.
- Projektordner `s4-plf-001_online_shop` mit IntelliJ öffnen.

1. Aufgabe - Product

Implementieren Sie die Model-Klasse `Product` gemäß UML-Diagramm und Klassenbeschreibung. Sie können die vorhandenen Testklassen für JUnit-Tests verwenden.

Table 1. Klassenbeschreibung

<table>
<tr><th>Attribut/Methode</th><th>Beschreibung</th></tr>
<tr><td><code>idCounter</code></td><td>

Counter, der für jede erzeugte <code>Product</code>-Instanz eine eindeutige Id vergibt.
Der Counter soll beim Wert 34 starten.

</td></tr>
<tr><td><code>id</code></td><td>

Jede <code>Product</code>-Instanz muss vom <code>idCounter</code> eine eindeutige Id erhalten, die nach ihrere Zuweisung nicht mehr verändert werden darf.

</td></tr>
<tr><td><code>name</code></td><td>

Repräsentiert den Namen eines Produktes.

</td></tr>
<tr><td><code>price</code></td><td>

Preis des Produktes in Euro.

</td></tr>
<tr><td><code>category</code></td><td>

Die Kategorie eines Produktes darf folgende Werte haben:

<table>
<tr><th>char</th><th>Kategorie-Bezeichnung</th></tr>
<tr><td><code>m</code></td><td>Media</td></tr>
<tr><td><code>f</code></td><td>Food</td></tr>
<tr><td><code>c</code></td><td>Cosmetics</td></tr>
<tr><td><code>o</code></td><td>Office Equipment</td></tr>
</table>

Wird ein falscher Wert übergeben, so soll eine <code>IllegalArgumentException</code> geworfen werden.

</td></tr>
<tr><td><code>toString</code>
<code>String</code></td><td>

Die Methode gibt einen String über sämtliche Informationen eines Produktes zurück:

<pre><code>ID, NAME, PRICE €, CATEGORY</code></pre>

Die Kategorie soll dabei ausgeschrieben werden.

Beispiel:
<pre><code>34, Smo-King Kaltrauchgenerator Big-Grill-SMO, 139.00 €, Food
35, WURSTBARON® Premium Tomahawk Steak vom Jungbullen, 49.99 €, Food
36, NYX Professional Makeup Epic Ink Eye Liner, 10.02 €, Cosmetics
37, Smoofl Hundeeis, Premium Hundesnacks, 13.46 €, Food
38, Bleistiftanspitzer Miauende Katze, 15.94 €, Office Equipment</code></pre>
</td></tr>
</table>

2. Aufgabe - OnlineShop

Implementieren Sie die Klasse `OnlineShop` gemäß UML-Diagramm und Klassenbeschreibung. Sie können die vorhandenen Testklassen für JUnit-Tests verwenden.

Table 2. Klassenbeschreibung

Attribut/Methode	Beschreibung
`name`	Der Name des Onlineshops.
`products`	• Alle Produkte sollen im Array `products` vom Typ `Product[]` gespeichert werden. • Insgesamt sollen **5** Produkte gespeichert werden. • Achten Sie **genau** darauf, dass keine Methode eine Referenz auf das Array zurückliefert.
`add( product )` *`boolean`*	• Die Methode fügt ein neues Produkt in den Onlineshop hinzu. • Das Produkt darf nicht doppelt aufgenommen werden. • Wenn ein Produkt aufgenommen wurde, so soll die Methode `true` zurückliefern. • Im Fehlerfall bzw. wenn ein Produkt nicht aufgenommen wurde, so soll die Methode `false` zurückliefern.
`sellById( id )` *`Product`*	• Die Methode dient zum Verkaufen eines Produktes. • Wenn das Produkt mit der übergebenen `id` im Shop vorhanden ist, so soll es aus dem Array entfernt werden. Geben Sie das entfernte Objekt zurück. • Ist das Produkt mit der gesuchten `id` nicht vorhanden, so soll der Wert `null` zurückgegeben werden.
`countProducts()` *`int`*	• Zählt alle im Shop vorhandenen Produkte und liefert den Wert zurück.
`sortByPrice()` *`Product[]`*	• Sortiert alle Produkte, **aufsteigend**, nach deren Preis. • Die Methode soll ein **neues** Array mit den sortierten Werten zurückliefern. • Alle `null` Werte sollen an das Ende des Arrays sortiert werden. • Das bestehende Array `products` soll unsortiert bleiben. **Hinweis** Kopieren Sie die Daten vor dem Sortieren zuerst in ein **neues** Array mit der gleichen Größe wie `products`.

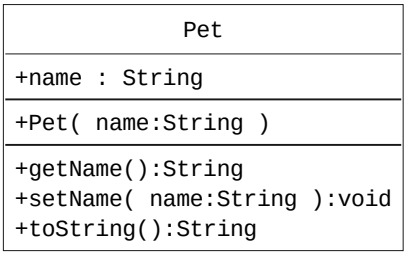
`toString()`
`String`
	• Liefert detaillierte Informationen über den Shop zurück:
	
	`Shopbezeichnung: Joe's Milchbar`
	`Vorhandene Produkte:`
	`34, Smo-King Kaltrauchgenerator Big-Grill-SMO 1,25 Liter, 139.00 €, Food`
	`...`
	
	• Verwenden Sie eine Instanz der Klasse `StringBuilder` um die Informationen der Produkte zu sammeln.

Attribut/Methode	Beschreibung
`BONUSAUFGABE` `sortByPrice( direction )`	• Erweitern Sie die Methode `sortByPrice()` um den Parameter `direction`. • Wenn ein falscher Wert übergeben wird, muss eine `IllegalArgumentException` geworfen werden. • Der Parameter `direction` gibt die Sortierreihenfolge an: **Aufsteigend** ◦ Wenn der Wert gleich wie dem Wert der Konstante `DIRECTION_ASC` ist. ◦ Wenn ein ungültiger Wert übergeben wurde. **Absteigend** ◦ Wenn der Wert gleich dem Wert der Konstante `DIRECTION_DESC` ist. • Alle `null` Werte sollen bei beiden Varianten an das Ende des Arrays sortiert werden.

3. Aufgabe - JUnit

Gegeben sei die Klasse `Pet` im package `test`. Implementieren Sie die JUnit Testklasse `PetTest` im richtigen Ordner.

- Analysieren Sie vor der Implementierung die Klasse `Pet`.
- Implementieren Sie **einen** JUnit-Testfall für die Methode `setName` in der Testklasse `PetTest`, der **nicht** die Funktion testet.



4. Aufgabe - Abgabeverfahren

- Projektordner `s4-plf-001_online_shop` mit dem Windows Explorer öffnen.
- Datei `abgabe.cmd` ausführen.
- **Z: Laufwerk** öffnen und Abgabe kontrollieren.
- Vom Prüfungsrechner **abmelden**.

Viel Erfolg!!!