Testat 2 der Vorlesung Programmierung

In diesem Testat sollen Sie die Klassen **Fridge, FridgeItem, Cheese** und **Yoghurt** anpassen. Die Schnittstelle **Weightful** ist vorgegeben und darf nicht verändert werden. Die Klasse **Main** enthält eine main-Methode, mit der Sie Ihre Implementierung testen können.

Aufgabe 1 (60 Punkte)
Ergänzen Sie die Klasse **FridgeItem** wie folgt:

- Fügen Sie der Klasse 3 private Instanzvariablen hinzu: "weight" vom Typ double, "name" vom Typ String und "color" vom Typ String.
- Initialisieren Sie diese Felder im vorgegebenen Konstruktor mit den Werten, die dieser als Parameter erhält.
- Implementieren Sie die Schnittstelle **Weightful** und geben Sie in der zugehörigen Methode den Wert der Instanzvariable "weight" zurück.
- Überschreiben Sie die toString() Methode aus java.lang.Object, so dass diese eine Zeichenkette in folgendem Format zurückgibt: <name> (<color> <weight> g) (also: Wert der Instanzvariable name, Leerzeichen, Klammer auf, Wert der Instanzvariable color, Leerzeichen, Wert der Instanzvariable weight, Leerzeichen, kleines g, Klammer zu)

Ergänzen Sie die Klassen **Cheese** und **Yoghurt** wie folgt:

- Ändern Sie die Vererbungsbeziehungen so, dass beide Klassen von FridgeItem erben.
- Fügen Sie den Klassen den Konstruktor Ihrer neuen Superklasse hinzu und geben Sie die Parameter an die Superklasse weiter.
- Überschreiben Sie die toString() Methode, so dass vor den Wert der Implementierung aus der Superklasse noch der Name der Klasse gefolgt von einem Doppelpunkt und einem Leerzeichen angefügt wird. D.h. analog zu FridgeItem soll die Ausgabe wie folgt aussehen:
  - Cheese: <name> (<color> <weight> g) (für toString() in Cheese)
  - Yoghurt: <name> (<color> <weight> g) (für toString() in Yoghurt)

Nachdem Sie diese Änderungen gemacht haben, sollten alle Kompilierungsfehler beseitigt sein.

Aufgabe 2 (40 Punkte)
Die Klasse **Fridge** (Kühlschrank) speichert eine begrenzte Menge an **FridgeItem** (Gegenständen) mit Hilfe eines "item"-Arrays. Die containsItem-Methode prüft, ob sich ein Gegenstand bereits im Kühlschrank (d.h. im Array) befindet. Die addItem-Methode fügt einen Gegenstand hinzu, sofern dieser nicht null ist und es noch Platz gibt (also items.length < maxItems). Erweitern Sie die Klasse wie folgt:

- Implementieren Sie die Schnittstelle Weightful und geben Sie über die getWeight-Methode die Summe der Gewichte aller Gegenstände im Kühlschrank zurück.
- Ergänzen Sie die Prüfungen in der addItem-Methode so dass diese zusätzlich noch sicherstellt, dass ein Gegenstand nur dann hinzugefügt wird.
  - wenn er sich noch nicht im Kühlschrank befindet (Tipp: containsItem)
  - wenn dadurch das maximale Gewicht (gespeichert in der Instanzvariable "maxWeight") nicht überschritten wird (Tipp: getWeight).
- Implementieren Sie die Methode indexOf(name,startIndex), sodass diese das FridgeItem mit dem Namen "name" im items-Array ab der Stelle "startIndex" bis "items.length - 1" sucht. Wird ein FridgeItem mit dem Namen gefunden, soll der kleinste Index zurückgegeben werden. Ist der Name nicht im Bereich von "startIndex" bis "items.length - 1" enthalten, soll -1 zurückgegeben werden. **Einschränkung:** Nutzen Sie für die Implementierung dieser Methode **keinerlei Schleifen, sondern ausschließlich Rekursion.**

Testat 2 der Vorlesung Programmierung

In diesem Testat sollen Sie die Klassen **MediaLibrary, MediaItem, Movie** und **Series** anpassen. Die Schnittstelle **Lasting** ist vorgegeben und darf nicht verändert werden. Die Klasse **Main** enthält eine main-Methode, mit der Sie Ihre Implementierung testen können.

Aufgabe 1 (60 Punkte)
Ergänzen Sie die Klasse **MediaItem** wie folgt:

- Fügen Sie der Klasse 3 private Instanzvariablen hinzu: "duration" vom Typ int, "title" vom Typ String und "views" vom Typ int.
- Initialisieren Sie diese Felder im vorgegebenen Konstruktor mit den Werten, die dieser als Parameter erhält.
- Implementieren Sie die Schnittstelle **Lasting** und geben Sie in der zugehörigen Methode den Wert der Instanzvariable "duration" zurück.
- Überschreiben Sie die toString() Methode aus java.lang.Object, sodass diese eine Zeichenkette in folgendem Format zurückgibt: <title> (<views> <duration> min)
  (also: Wert der Instanzvariable title, Leerzeichen, Klammer auf, Wert der Instanzvariable views, Leerzeichen, Wert der Instanzvariable duration, Leerzeichen, kleines m, kleines i, kleines n, Klammer zu).

Ergänzen Sie die Klassen **Movie** und **Series** wie folgt:

- Ändern Sie die Vererbungsbeziehungen so, dass beide Klassen von MediaItem erben.
- Fügen Sie den Klassen den Konstruktor ihrer neuen Superklasse hinzu und geben Sie die Parameter an die Superklasse weiter.
- Überschreiben Sie die toString() Methode, sodass vor den Wert der Implementierung aus der Superklasse noch der Name der Klasse gefolgt von einem Doppelpunkt und einem Leerzeichen angefügt wird. D.h. analog zu MediaItem soll die Ausgabe wie folgt aussehen:
  - Movie: <title> (<views> <duration> min) (für toString() in Movie)
  - Series: <title> (<views> <duration> min) (für toString() in Series)

Nachdem Sie diese Änderungen gemacht haben, sollten alle Kompilierungsfehler beseitigt sein.

Aufgabe 2 (40 Punkte)
Die Klasse **MediaLibrary** speichert eine begrenzte Menge an **MediaItem** mit Hilfe eines "item"-Arrays. Die containsItem-Methode prüft, ob sich ein MediaItem bereits in der MediaLibrary (d.h. im itemsArray) befindet. Die addItem-Methode fügt ein MediaItem hinzu, sofern dieses nicht null ist und es noch Platz gibt (also items.length < maxItems). Erweitern Sie die Klasse wie folgt:

- Implementieren Sie die Schnittstelle **Lasting** und geben Sie über die getDuration-Methode die Summe der Dauer aller MediaItems in der MediaLibrary zurück.
- Ergänzen Sie die Prüfungen in der addItem-Methode, sodass diese zusätzlich noch sicherstellt, dass ein MediaItem nur dann hinzugefügt wird
  - wenn es sich noch nicht in der MediaLibrary befindet (Tipp: containsItem)
  - wenn dadurch die maximale Dauer (gespeichert in der Instanzvariable "maxDuration") nicht überschritten wird (Tipp: getDuration).
- Implementieren Sie die Methode lastIndexOf(title,startIndex) so dass diese das MediaItem mit dem Titel "title" im items-Array ab der Stelle "startIndex" bis 0 sucht. Wird ein MediaItem mit dem Titel gefunden, soll der höchste Index zurückgegeben werden. Ist der Name nicht im Bereich von "startIndex" bis 0 enthalten, soll -1 zurückgegeben werden. **Einschränkung:** Nutzen Sie für die Implementierung dieser Methode **keinerlei Schleifen, sondern ausschließlich Rekursion.**