PROO/SEW 3 - Test: Schulbibliothek

Ziel

Implementierung eines simplen Verwaltungsprogramms für die Schulbibliothek.

Lernziele

- Einsatz des *Factory*-Entwurfsmusters
- Verwenden verschiedener *Java-Collections*
- Schreiben von *Log*-Einträgen

Materialien

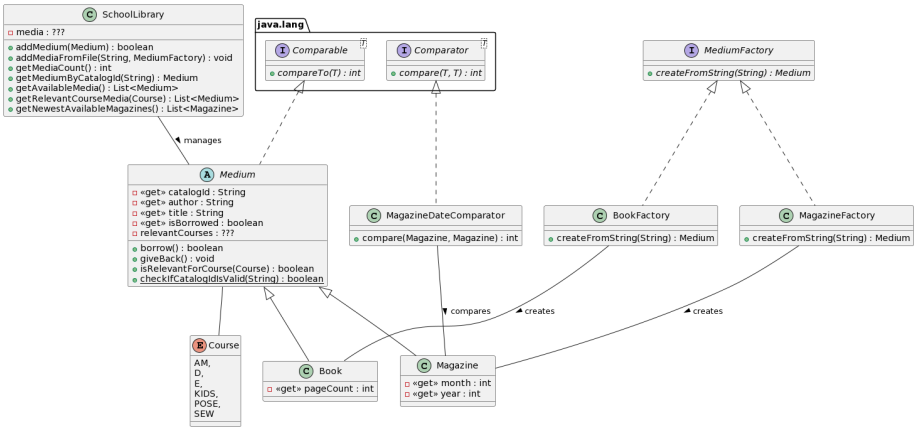
- *IntelliJ*

Abgaberichtlinien

- Die fertige *IntelliJ*-Lösung ist in einem Ordner mit ihrem **Nachnamen (ohne Sonderzeichen)** auf dem `U:`-Laufwerk abzugeben.

Aufgabe

Sie wurden von der Schulbibliothek mit der Implementierung einer *Java*-Lösung zur Verwaltung von Leihmedien beauftragt. Aktuell werden lediglich Bücher und Magazine verliehen - allerdings soll das Sortiment in Zukunft auch um Filme und *CDs* erweitert werden. Um möglichst hohe Erweiterbarkeit zu gewährleisten, entscheiden Sie sich das *Factory-Pattern* einzusetzen und skizzieren folgendes Klassendiagramm:



Abseits von den zur Verfügung gestellten *Unit-Tests* erhalten Sie folgende Liste an Anforderungen:

- Jedes `Medium` besteht aus einer eindeutigen Katalognummer, einem/r AutorIn (bzw. Verlag), einem Titel und einer *Menge* an relevanten Schulfächern.
  - Mittels `isRelevantForCourse` wird überprüft, ob ein Medium relevant für ein bestimmtes Schulfach ist.
  - Für die Katalognummer wurde ein einfacher Prüfziffer-Algorithmus konzipiert (s.u.). Wird eine ungültige Katalognummer angegeben, so ist eine `IllegalArgumentException` zu werfen.
    - Die Katalognummer wird weiters verwendet, um die *Gleichheit* von zwei Medien zu überprüfen!
  - Zwei simple Methoden ermöglichen das Ausleihen und Zurückgeben von Medien. Mittels `borrow` kann ein nicht-ausgeliehenes Medium ausgeliehen werden, wobei der Erfolg mittels *booleschem* Rückgabewert signalisiert wird. Mittels `giveBack` kann dieses auch zurückgegeben werden - wird allerdings ein nicht-ausgeliehenes Medium zurückgegeben, ist eine eigene `SchoolLibraryException` zu werfen!
  - Um Ordnung in den Regalen der Schulbibliothek zu wahren, werden Medien zuerst anhand deren AutorIn, dann anhand deren Titel aufsteigend sortiert.
- Bücher *erweitern* die Funktionalität von Medien zusätzlich um die Seitenzahl. Wird eine Seitenzahl kleiner als 1 angegeben, so ist eine `IllegalArgumentException` zu werfen.
- Magazine *erweitern* die Funktionalität von Medien um einen Monat (zwischen 1 und 12, sonst ist eine `IllegalArgumentException` zu werfen!) und ein Jahr.
  - Zusätzlich soll der `MagazineDateComparator` das *absteigende* Sortieren nach Erscheinungsdatum ermöglichen.
- Zwei *Factories* sollen vom *Interface* `MediumFactory` ableiten und die Anlage von Büchern und Magazinen anhand von Komma-separierten Zeilen ermöglichen.
- Die `SchoolLibrary` verwaltet Medien in einer sinnvoll gewählten *Collection* Ihrer Wahl und bietet verschiedene Methoden für den Zugriff auf diese.
  - Mittels `addMedium` können Medien hinzugefügt werden, wobei die Katalognummern eindeutig sein müssen. Wird eine Katalognummer wiederverwendet ist das Medium nicht zu überschreiben und `false` zurückzugeben!
  - `addMediaFromFile` erlaubt das Hinzufügen von Medien aus `.csv`-Dateien unter Zuhilfenahme einer als Parameter übergebenen *Factory*. Bei falschen Pfaden oder sonstigen Zugriffsproblemen ist die `IOException` in die eigene `SchoolLibraryException` zu verpacken.
  - `getMediumByCatalogId` liefert ein bestimmtes Medium anhand dessen Katalognummer.
  - Die `getAvailableMedia`-Methode retourniert alle nicht-ausgeliehenen Medien, sortiert nach AutorIn und Titel.
  - `getRelevantCourseMedia` liefert alle (auch ausgeliehene!) Medien die für einen bestimmten Kurs relevant sind, sortiert nach AutorIn

und Titel.
  - Mittels `getNewestAvailableMagazines` können die neuesten, nicht-ausgeliehenen Magazine nach Erscheinungsdatum sortiert geholt werden.
- Erweitern Sie die *Factories* und die `SchoolLibrary`-Methoden zum Hinzufügen von Medien um *Logging*, so dass der Programmablauf nachvollziehbarer wird und Fehler beim Laden aus den `.csv`-Dateien leichter gefunden werden können.

Prüfziffer-Berechnung

Jede Katalognummer besteht aus der Zeichenkette `LEO` gefolgt von einer Zahl. Alle Ziffern werden mit einer Gewichtung - die erste Ziffer mit 3, die zweite Ziffer mit 4 usw. - multipliziert und summiert. Die Summe *modulo* 10 ergibt dann die Prüfziffer, wie folgendes Beispiel für die Katalognummer *LEO96685?* demonstriert:

Zahl	LEO	9	6	6	8	5	
Gewichtungen		3	4	5	6	7	
Produkte		27	24	30	48	35	
Summe							164
Summe *mod* 10							**4**

Die Prüfziffer ist also 4, wodurch die vollständige, korrekte Katalognummer *LEO966854* ist.