Babeș-Bolyai Universität, Cluj Napoca
Fakultät für Mathematik und Informatik
Objekt-Orientierte Programmierung, 2021

Labor 5

Diskussion: 26./27./28. April, **Abgabe:** 9. April, **Wert:** 10/50

Teilnahme am Labor wird nur für diejenigen registriert, die etwas vorzeigen!

Die Laboraufgabe ist in Teams von 3 Personen zu bearbeiten.

In den folgenden Laboraufgaben wird die Anwendung zu erweitern sein!

Die Anwendung soll einem Gebrauchtwagengeschäft zur Verwaltung des Lagers dienen.
Jedes Auto wird durch die folgenden Charakteristika beschrieben: Modell, Marke, Erstzulassungsjahr, Kilometer, Preis, Leistung (PS), Treibstoff.

Funktionale Anforderungen

F1. Das Programm soll von 2 unterschiedlichen Benutzern verwendet werden: Manager und Kunde. Zu Beginn des Programms wählt der Benutzer seine Rolle aus.
F2. Ein Manager kann Autos hinzufügen, löschen oder bearbeiten können.
F3. Ein Kunde oder Manager möchte auch nach Autos suchen können, Modell/Marke.
F4. Ein Kunde oder Manager möchte die Autoliste filtern, nach dem Alter/Kilometer, einem vom Benutzer angegebenen Wert.
F5. Ein Kunde oder Manager möchte alle Autos nach Preis aufsteigend sortiert sehen.
F6. Ein Kunde soll Autos aus der Liste zu seinen persönlichen Favoriten hinzufügen können, und eine Liste seiner Favoriten ausgeben können.

Weitere Anforderungen

W1. Die Interaktion des Benutzers mit dem Programm passiert via **Konsole**. Dem Benutzer werden dort das Menu und Ergebnissen angezeigt, und dort auch die Befehle des Benutzers eingelesen (User Interface, UI).
W2. Das Programm soll bei Start Beispieldaten von 10 oder mehr Produkten beinhalten. Die Lagerdaten müssen **nicht persistent** gespeichert werden.
W3. Ausnahmen (z.b. ungültige Eingaben) sollen sorgfältig behandelt werden.

Qualitäts-Anforderungen

Q1. Die Anwendung soll in einer Schicht-Architektur (Labor 4 – Beispiel) angelegt sein, und entsprechend **modularisiert** werden (mit namespaces für domain, repository, controller, ui);
Q2. Wo Zeiger notwendig sind, dort sind **Smart-Pointer** zu verwenden.
Q3. Code sollte **selbsterklärend** sein: Klassen, Methoden und Variablen sollen **zweckgemäße Namen** haben.

Abgabekriterien

K1. Die Lösung kann aus mehreren Dateien bestehen. Diese sind als ZIP abzugeben **`L5_TeamX.zip (X ist die Nummer)`** (kein RAR, 7z, o.ä., das Archiv beinhaltet nur Quellcode und keine Verzeichnisse)
K2. Die Datei **`Main.cpp`** enthält `main()`
K3. Das Programm muss mit dem GnuCompiler auf folgende Weise kompilierbar sein:
```
g++ -std=c++17 -o prog *.cpp
```

Empfehlungen

R1. Implementierung der Module in der Reihenfolge ihrer Verwendung, also aus der Perspektive des Benutzersicht: domain, ui, controller, repository.
R2. Um sicherzustellen dass repository und controller funktionieren, ohne Verwendung der Benutzeroberfläche, sind Unit-Tests nützlich;