Babeș-Bolyai Universität, Cluj Napoca
Fakultät für Mathematik und Informatik
Objekt-Orientierte Programmierung, 2023

Labor 4

Diskussion: Kalenderwoche 17, **Abgabe:** Kalenderwoche 18, **Wert:** 6/50

Zum Diskussionstermin muss zumindest die Anwendungsstruktur definiert sein, ansonsten wird die Laborteilnahmen nicht gewertet!

Klaus ist der Manager eines Früchteladens, und er braucht eine Anwendung, um sein Produktelager zu verwalten. Klaus hat mit Hilfe von einem Analysten eine Liste von Anforderungen erstellt.

Funktionale Anforderungen

F1. Für jedes Produkt werden folgende Attribute gespeichert: Name, Herkunft, Haltbarkeitsdatum, Menge und Preis.
F2. Er möchte Produkte hinzufügen oder löschen können. Jedes Produkt wird durch seinen Namen und seinen Hersteller eindeutig identifiziert. Falls ein vorhandenes Produkt hinzugefügt wird, wird nur seine Menge aktualisiert.
F3. Er möchte alle verfügbaren Produkte auflisten, die eine bestimmte Zeichenkette (String) enthalten (falls der String leer ist, werden alle verfügbaren Produkte zurückgegeben), sortiert nach Namen.
F4. Er möchte nur die Produkte sehen, die knapp sind (Menge kleiner als ein vom Benutzer angegebener Wert).
F5. Er möchte alle Produkte nach Haltbarkeitsdatum aufsteigend sortiert sehen.

Weitere Anforderungen

W1. Die Interaktion des Benutzers mit dem Programm passiert via Konsole (User Interface, UI). Dem Benutzer werden dort das Menu und Ergebnissen angezeigt, und dort auch die Befehle des Benutzers eingelesen.
W2. Das Programm soll bei Start Beispieldaten von zumindest 10 Produkten beinhalten. Die Lagerdaten müssen nicht persistent gespeichert werden.

Qualitäts-Anforderungen

Q1. Die Anwendung soll dem untenstehende Architekturbeispiel folgen, und entsprechend **modularisiert** werden (mit namespaces für domain, repository, controller, ui);
Q2. Wo Listen notwendig sind, dort sind **Container** aus der STL zu verwenden, und wo Zeiger notwendig sind, dort sind **Smart-Pointer** zu verwenden.
Q3. Code sollte **selbsterklärend** sein: Klassen, Methoden und Variablen sollen **zweckgemäße Namen** haben (in englisch!) und einer einheitlichen Namenskonvention folgen.

Abgabekriterien

K1. Die Lösung kann aus mehreren Dateien bestehen. Diese sind als ZIP abzugeben **`L3_Nachname_Vorname.zip`** (kein RAR, 7z, o.ä., das Archiv beinhaltet nur Quellcode und keine Verzeichnisse)
K2. Die Datei **`Shop.cpp`** enthält die `main`-Funktion.
K3. Das Programm muss mit dem GnuCompiler auf folgende Weise kompilierbar sein:
```
g++ -std=c++20 -o shop.exe *.cpp
```

Empfehlungen

R1. Studiere die untenstehende Anwendungsarchitektur genau, und definiere die Struktur deines Programms in ähnlicher Weise.
R2. Implementierung der Module in der Reihenfolge ihrer Verwendung, also aus der Perspektive des Benutzersicht: `domain, ui, controller, repository`.

Babeș-Bolyai Universität, Cluj Napoca
Fakultät für Mathematik und Informatik
Objekt-Orientierte Programmierung, 2023

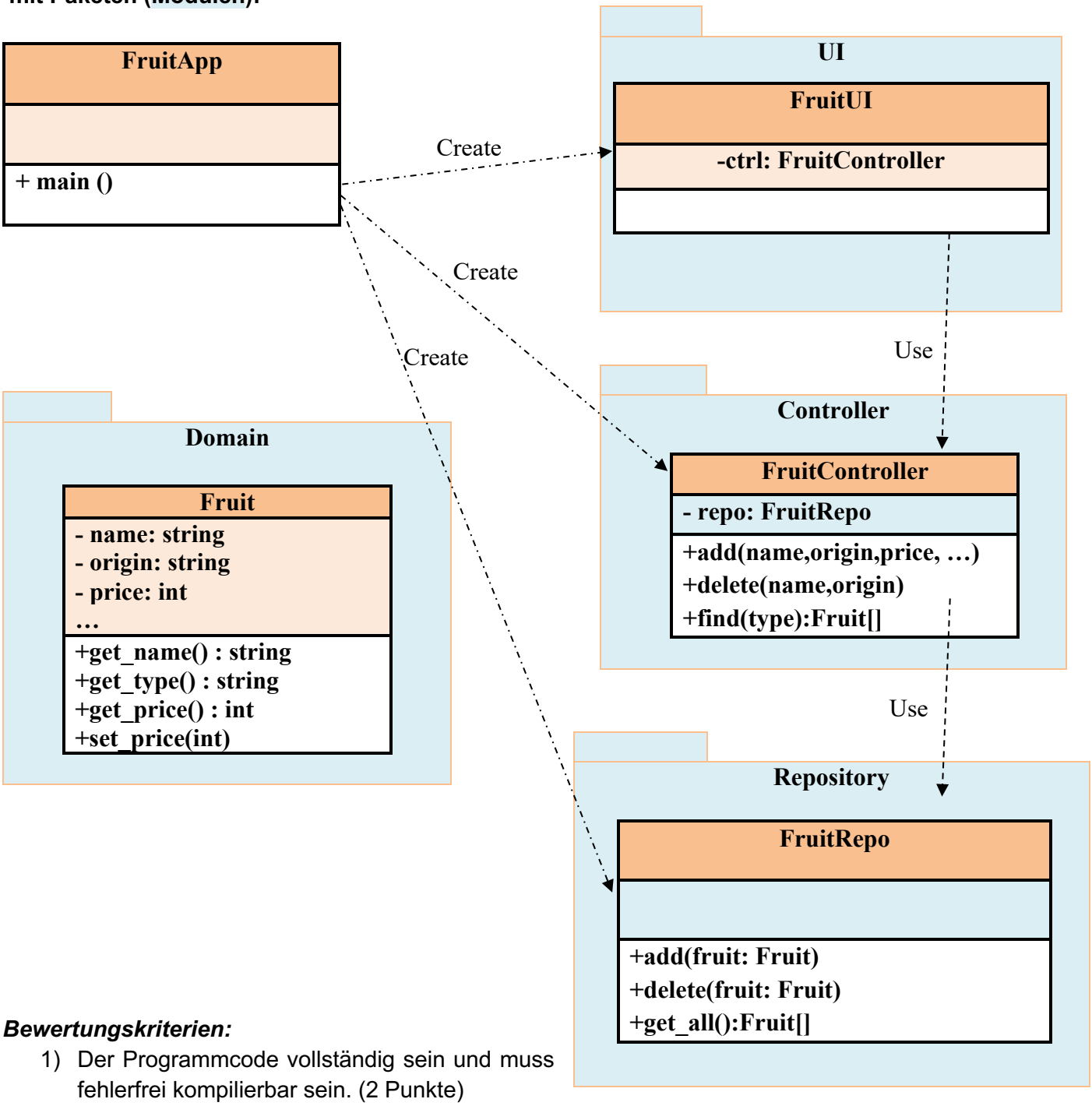
Labor 4

Diskussion: Kalenderwoche 17, **Abgabe:** Kalenderwoche 18, **Wert:** 6/50

Zum Diskussionstermin muss zumindest die Anwendungsstruktur definiert sein, ansonsten wird die Laborteilnahmen nicht gewertet!

R3. Um sicherzustellen, dass `repository` und `controller` funktionieren, ohne Verwendung der Benutzeroberfläche, sind Unit-Tests nützlich;

Architektur in Form eines UML-Klassendiagrams mit Paketen (Modulen):



Bewertungskriterien:

1) Der Programmcode vollständig sein und muss fehlerfrei kompilierbar sein. (2 Punkte)
2) Das Programm muss ausführbar sein und das korrekte Ergebnis liefern. (2 Punkte)
3) Du musst den Code erklären können. (4 Punkte)
4) Der Code muss gut lesbar sein. (2 Punkte)