Herzlichen Glückwunsch!

Der Vertrieb hat ein neues Projekt an Land gezogen. Es handelt sich um die Weiterentwicklung einer bestehenden Anwendung zum Verwalten von Studenten. Dabei gibt es eine gute und eine schlechte Nachricht:

Die gute Nachricht ist, dass die Anwendung recht überschaubar ist und über automatisierte Unit Tests verfügt.

Nun die schlechte Nachricht: Trotz der kleinen Codebasis scheint die Implementierung nicht besonders einfach erweiterbar zu sein, da der Code nicht sehr flexibel ist und von den bisherigen Entwicklern vieles fest verdrahtet wurde. Leider sind diese Entwickler nicht mehr verfügbar, so dass wir die Weiterentwicklung selbst übernehmen müssen.

Über kurz oder lang erwarten wir weitere Anforderungen vom Kunden. Es besteht die Gefahr, dass Weiterentwicklungen durch die genannten technischen Schulden komplizierter und teurer werden als nötig.

Zudem sieht es so aus, als ob die vorhandenen Unit Tests sowohl quantitativ als auch qualitativ verbesserungswürdig sind.

Systembeschreibung

***Hinweis**:* *Das ist eine Sollbeschreibung der Anforderungen - wir sind nicht sicher, ob das System in Wirklichkeit genauso funktioniert!*
Genauso ist nicht vollkommen klar, ob alle diese Anforderungen vollständig und widerspruchsfrei sind.

Das System besteht aus den Klassen `Main`, `Student`, `Address`, `PhoneNumber` und `DataStore`. Daneben gibt es in einem zweiten Source-Ordner namens `test` noch Testklassen mit einigen Unit Tests.

Die Klasse **`Main`** steuert die Anwendung. Sie ist als textbasierte Konsolenanwendung ausgeführt und bietet dem Benutzer alle Möglichkeiten, um Studenten zu suchen und ihre Daten anzuzeigen.

Die Klasse **`Student`** ist das Zentrum der Anwendung. Sie implementiert die Funktionalitäten, die von der Klasse `Main` gesteuert werden.

Im Wesentlichen handelt es sich dabei um die folgenden Möglichkeiten:

- nach Studenten anhand ihrer Matrikelnummer (ID) zu suchen
- eine einfache Info anzuzeigen (bestehend aus ID, Vorname und Name)
- die Adresse und die Telefonnummer anzuzeigen
- die Anwendung zu beenden

Wird dabei nach einer Matrikelnummer gesucht, die das System nicht kennt, soll eine Fehlermeldung ausgegeben werden und erneut gesucht werden können.

Da die Anwendung bisher nur in Deutschland verkauft wurde, sind die Daten (Adresse und Telefonnummer) bei der Ausgabe so formatiert, wie es in Deutschland üblich ist:

- Adresse: Straße [Leerzeichen] Hausnummer [Neue Zeile] PLZ [Leerzeichen] Ort
- Telefon: Vorwahl [Schrägstrich] Nummer

Für die Formatierung der Daten sind zwei Hilfsklassen in die Anwendung integriert: **`Address`** und **`PhoneNumber`**, die beide dazu dienen, die Adress- und Telefondaten eines Studenten für die Bildschirmausgabe so zu formatieren, wie ein Benutzer sie erwarten würde.

***Hinweis**: Weiters gibt es noch die Klasse **`DataStore`**, die für die Aufgabenstellung eine Datenbank simuliert, und es der Anwendung ermöglicht, Daten von Studenten aus einer Textdatei (datastore.csv) zu laden. Diese Klasse soll nicht Gegenstand der Betrachtung sein. Natürlich darf die Textdatei, insbesondere für Testzwecke, bearbeitet und erweitert werden.*

Weiterentwicklung

In Zukunft erwarten wir durch die Globalisierung verstärkte Möglichkeiten, unser wunderbares Studentenverwaltungssystem in andere Länder verkaufen zu können. Besonders aus Frankreich, Großbritannien und den USA liegen uns bereits Anfragen vor.

Natürlich erwarten die Kunden aus diesen Ländern, dass die Adressen und Telefonnummern so dargestellt werden, wie es vor Ort üblich ist.

Die Anwendung muss also in der nächsten Version die Möglichkeit bieten, in Abhängigkeit vom Land über irgendeine Art von Konfiguration die Art und Weise zu steuern, wie Adress- und Telefondaten dargestellt werden.

Es ist ausreichend, wenn die Anwendung zur Laufzeit *eine* Art von Formatierung unterstützt. Wir wollen nur bei der erstmaligen Konfiguration festlegen können, *welche* Art der Formatierung wir verwenden. Die Installation soll dabei in jedem unterstützten Land ohne die Anpassung von Quellcode möglich sein.

Dabei ist zu beachten, dass keine Mischkonfigurationen möglich sein sollen (z.B. Adresse wie in USA, Telefonnr wie in Deutschland) – die Konfiguration muss zuverlässig und konsistent die gesamte Familie von Darstellungen umschalten.

Es ist zu erwarten, dass wir in absehbarer Zeit noch einige weitere Länder mit dem System beliefern können. Daher wäre es gut, wenn neue Länder möglichst einfach hinzugefügt werden können.

Aufgabenbeschreibung

1. Bringe zunächst das System zum Laufen und finde heraus, wie es funktioniert

2. Überprüfe durch manuelles Testen am laufenden System, ob alle Anforderungen implementiert sind wie oben beschrieben

3. **Dokumentiere** sowohl **Einhaltung** als auch **Abweichungen** von den beschriebenen Anforderungen
   → *Dokumentation wird bewertet (Inhalt, nicht Form)*

4. Lass die Unit Tests laufen, bzw. bringe sie zum Laufen (d.h. fixe vor der Weiterentwicklung die bestehenden Unit Tests, falls notwendig)

5. **Ergänze** falls notwendig die vorhandenen **Tests**, so dass alle Anforderungen über Unit Tests abgedeckt sind (alle Unit Tests müssen **erfolgreich durchlaufen**)
   → *Testabdeckung und Lauffähigkeit der Tests werden bewertet*

6. Implementiere die oben beschriebene Weiterentwicklung. **Verwende** dafür eines oder mehrere **passende Entwurfsmuster**, die dafür sorgen, dass die kleine Produktfamilie (Adresse, Telefonnummer) in Abhängigkeit vom Einsatzort der Anwendung in der jeweils passenden Variante erzeugt wird. **Begründe** deine **Entscheidung** für (und/oder gegen) Entwurfsmuster
   → *Verwendung von Entwurfsmuster(n) und Dokumentation werden bewertet*

7. Stelle sicher, dass die **neue Funktionalität** durch **ausreichende Abdeckung** mit **lauffähigen Unit Tests** abgesichert und dokumentiert ist
   → *Neue Funktionalität, Testabdeckung und Lauffähigkeit der Tests werden bewertet*

8. Selbstverständlich muss die Anwendung **weiterhin lauffähig** sein
   → *Lauffähigkeit der Anwendung wird bewertet*

***Weitere Hinweise**:*

- *Softwareentwicklung ist ein sozialer Vorgang. Die besten Entwürfe und Implementierungen entstehen in Teamarbeit. Deshalb ist es **möglich**, die Aufgabe als **Gruppe** zu bearbeiten (maximal 3 Personen pro Gruppe)*
- *Alle Mitglieder einer **Gruppe** werden **dieselbe Note** für ihren **Programmentwurf** erhalten*
- *Verständnisfragen sind nicht verboten*
- *Hinweise auf Widersprüche oder Fehler in der Aufgabenstellung sind ausdrücklich erwünscht*

Viel Erfolg!