Lehrstuhl für Angewandte Informatik I
Prof. Dr. Bernhard Westfechtel
Sandra Greiner

WS 2017/18
09. November 2017

Konzepte der Programmierung

Übung 4 – Abgabe: 23.11.2016

Aufgabe 4.1 (Mobiltelefon – Fortsetzung)
Mit dieser Aufgabe setzen wir die Aufgabe von Blatt 3 fort und erlauben es, ein kleines Telefonbuch bestehend aus Rufnummern auf der SIM-Karte zu speichern. Dafür werden Arrays mit der Klasse interagieren bzw integriert.

(a) Fügen Sie der Klasse `SimKarte` ein neues **privates Objektfeld** `telefonbuch` hinzu. Darin sollen bis zu 5 Nummern (`String`) sequentiell abgelegt werden. Das Feld soll sofort den Speicher für 5 Zahlen reservieren.

(b) Für die sequentielle Speicherung müssen Sie sich pro `SimKarte` merken, an welcher Stelle die nächste Nummer gespeichert werden kann (oder wie viele Nummern schon vorhanden sind). Legen Sie an geeigneter Stelle einen *Zähler* an.

(c) Schreiben Sie eine `Objektmethode` `fuegeEin()` in der Klasse `SimKarte`, die eine ihr **übergebene Rufnummer** an der nächsten freien Stelle einfügt. Sollten schon 5 Telefonnummern gespeichert sein, wird eine Fehlermeldung auf der Konsole ausgegeben und die Methode bricht ab.

(d) Schreiben Sie auch eine Objektmethode `gibAus()`, die nichts *zurückgibt*, aber den Netzanbieter sowie den Inhalt des Telefonbuchs *auf der Konsole ausgibt*.

(e) Schreiben Sie nun einen Klasse `Main` mit einer main-Methode mit Standardsignatur. In der Methode soll zunächst ein *neues* SIM-Karten-Objekt erstellt werden. Dann sollen nacheinander 5 Rufnummern auf der SIM-Karte gespeichert werden und danach die Daten mithilfe von `gibAus()` auf der Konsole ausgegeben werden.
Fügen Sie in der main-Methode anschließend eine 6. Rufnummer hinzu. Erfüllt ihr Programm das erwünschte Verhalten?
Zuletzt fügen Sie die Karte in ein von Ihnen erzeugtes Mobiltelefon ein und schalten dieses programmatisch an.
Der Teil der Programmausgabe bis zum Einfügen der 6. Nummer könnte wie folgt aussehen:

```
SIM-Karte des Netzanbieters: vodafone
Telefonbuchinhalt:
0964234
123
7234
09958/4321
96
Das Telefonbuch der SIM Karte ist voll.
```

(2 + 1 + 4 + 3 + 4 = 14 Punkte)

Aufgabe 4.2 (Syntax für Sandwichs)
Ein Sandwich besteht aus zwei umgebenden Toastscheiben (`T`). Auf die untere Hälfte kann Mayonnaise (`M`) oder Senf (`S`) oder nichts gestrichen werden. Als nächste Schicht kommt immer Gemüse. Das kann eine Gurkenscheibe (`G`), Paprikastücke (`P`) oder ein Salatblatt (`SB`) sein. Danach folgt auf jeden Fall immer der Hauptbelag, der entweder aus Rindfleisch (`R`), Hähnchenfleisch (`H`) oder einer Falafel (`F`) besteht. Darauf kann Käse (`K`) folgen. Bei manchen Sandwichs folgt auf den ersten Hauptbelag nochmals eine Toastscheibe und eine zweite Scheibe Hauptbelag. Solche Sandwichs sind ohne Käse. In jedem Fall folgen dann noch optional beliebig viele Gemüseelemente. Als letzte Schicht vor der letzten Toastscheibe sind Zwiebeln (`Z`) möglich.

Das folgende Beispiel-Sandwich enthält zwischen den umgebenden Brötchenhälften Mayonnaise, ein Salatblatt, eine Scheibe Falafel mit Käse, eine Paprika, eine Gurke und nochmals eine Paprika und darauf Zwiebeln:

```
T M SB F K P G P Z T
```

(a) Geben Sie eine kontextfreie Grammatik in **BNF** (Kapitel Syntax, Folie 13ff.) zur „Produktion“ eines Sandwichs an.

(b) Geben Sie die Produktion des Sandwichs nun in der **EBNF** Notation an.

(5 + 3 = 8 Punkte)

Aufgabe 4.3 (Studentenverwaltung)
Wir beginnen die Entwicklung eines Programms zur Verwaltung von Studenten.

(a) Erstellen Sie dazu zunächst eine Klasse `Student`, die folgende Informationen von Studenten speichert:

- einen Nachname
- bis zu 4 Vornamen (in einem Array)
- MatrikelNr (fortlaufend)

Achten Sie dabei auf die Sichtbarkeiten Ihrer Objektfelder und überlegen Sie, wie Sie eine fortlaufende Nummerierung für jeden Studenten gewährleisten können.

(b) Erstellen Sie einen Konstruktor, der den Nachname und **einen** Vornamen erwartet. Die Matrikelnummer soll darin automatisch gesetzt werden und so agieren, dass die nächste Nummer zuverlässig weiter gezählt wird für alle Objekte. Der Vorname wird an die erste Position im Array geschrieben.

Überprüfen Sie Ihre Implementierung, indem Sie mehrere Objekte mittels BlueJ erzeugen und inspizieren.
Optional: Alternativ können Sie die Methode programmatisch testen. Dazu müssen Sie eine main-Methode in der Klasse `Student` hinzufügen und darin die Studenten anlegen.

(3 + 3 = 6 Punkte)