TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

Lehrstuhl für Sprachen und Beschreibungsstrukturen
Praktikum Grundlagen der Programmierung
Prof. Dr. Helmut Seidl

SS 2008
Aufgabenblatt 11
30. Juni 2008

Abgabe: 08.-10. Juli 2008 beim jeweiligen Tutor

Praktikum Grundlagen der Programmierung

Themen: Threads, Nebenläufige Programmierung

Aufgabe 50 (Ü) Ampelsteuerung

Implementieren Sie eine einfache Ampel, die die Abläufe an einem Fussgängerüberweg regelt. Die Ampel soll dabei den Fussgängern bzw. Autos eine gewisse Zeit die Berechtigung geben, die Strasse zu überqueren, bzw. den Fussgängerüberweg zu passieren.

a) Implementieren Sie zunächst einen Thread `Ampel`. Dieser habe folgende Eigenschaften:
   - Die Ampel habe nur zwei Zustände: Entweder können die Fussgänger gehen oder die Autos passieren.
   - Im Betrieb wechselt die Ampel permanent zwischen diesen Zuständen, wobei sie in jedem Zustand eine Minute bleibt.

b) Geben Sie die Implementierung eines Threads `Fussgaenger` an. Der Fussgänger sei durch einen Namen und die Ampel, an der er steht, gekennzeichnet. Er soll solange warten, bis die Ampel ihm das überqueren erlaubt.

c) Schreiben Sie eine `main`-Methode, die eine Ampel und zwei Fussgänger startet.

Aufgabe 51 (Ü) Paternoster

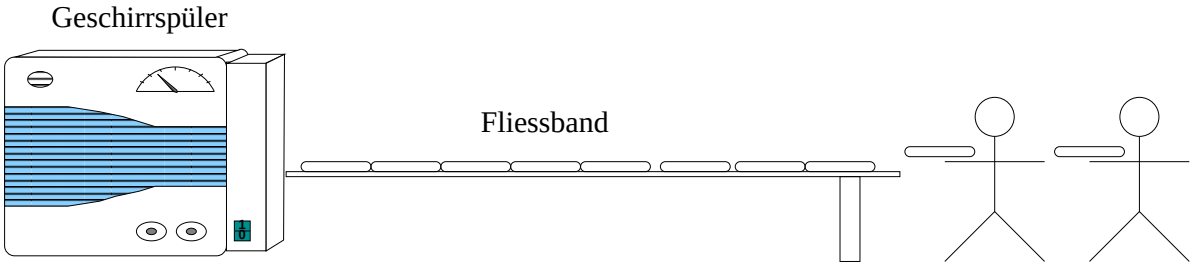
Ein Paternoster ist ein Aufzug, der ständig zwischen dem untersten und obersten Stockwerk hin und her pendelt. Die Fahrgäste können dabei den Aufzug nicht explizit anfordern, sondern müssen warten bis die Kabine ihr Stockwerk passiert, um einzusteigen.

Ziel dieser Aufgabe ist es, wesentliche Abläufe beim Betrieb eines solchen Aufzugs zu simulieren. Um die nebenläufig auftretenden Ereignisse in diesem Szenario umsetzen zu können, sollten dabei sowohl der Aufzug als auch die Fahrgäste als eigene Threads realisiert werden.

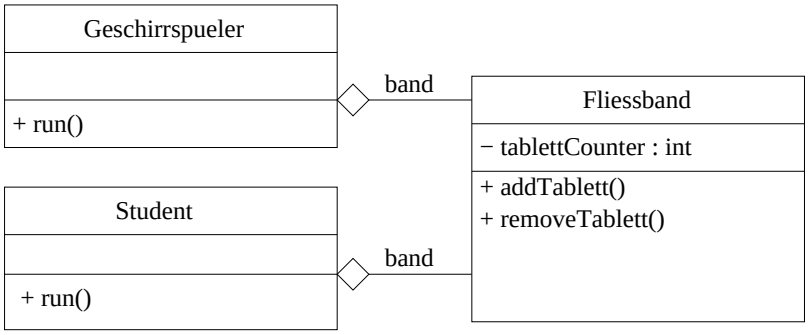
Fahrgäste stellen sich zu einem zufälligen Zeitpunkt an einem zufälligen Stockwerk am Paternoster an, um dann in einem zufälligen Stockwerk wieder auszusteigen. Die Kabine des Paternosters bewegt sich in unserer Simulation jede Sekunde um ein Stockwerk nach oben oder unten, um dann jeweils eine halbe Sekunde lang Passagiere aus- und einsteigen zu lassen.

a) Überlegen Sie sich den Ablauf der Simulation und v.a. die Synchronisation, die notwendigerweise zwischen den Threads ablaufen muss.
b) Entwerfen Sie ein Klassenmodell für die beiden Klassen `Paternoster` und `Fahrgast`.
c) Implementieren Sie die beiden Klassen und testen Sie Ihr Programm.

Aufgabe 52 (H) Mensa: Geschirrrückgabe **(12 Punkte)**



Der Mensa-Geschirrspüler ist ein Automat, der nach dem Essen die Tabletts von Studenten entgegennimmt und diese säubert. Damit nicht jeder Student darauf warten muss, dass der Geschirrspüler fertig wird, gibt es ein Fließband, auf das ein Student sein Tablett ablegen kann und die Mensa verlassen kann, ohne darauf zu warten, bis sein Tablett an der Reihe ist. Leider hat dieses Fließband nur Platz für 8 Tabletts, wenn es voll ist, müssen die restlichen Studenten warten...



a) Entwickeln Sie die Klasse `Fliessband` als Puffer zwischen Geschirrspüler und Studenten. Alle Methoden sollen dabei so verfasst werden, dass nebenläufige Prozesse sicher darauf zugreifen können. Das Fließband muss die Methoden `removeTablett` und `addTablett` bereitstellen:

   (i) `addTablett` Das Fließband darf nie mehr als 8 Tabletts akzeptieren. Studenten, die dann ankommen, müssen darauf warten, dass (z.B. durch den Geschirrspüler) ein Tablett entfernt, also ein Platz frei, wird.

   (ii) `removeTablett` soll dem Geschirrspüler ermöglichen, ein Tablett vom Fließband zu nehmen, sobald eines darauf liegt.

b) Entwickeln Sie die Klasse `Student` als Thread, der versucht, sein Tablett über das `Fliessband` dem `Geschirrspueler` zu übergeben.

HINWEIS: Sie dürfen bei der Implementierung die Modellierung von Tabletts weglassen. Es reicht bereits, wenn das Fließband einfach nur mitzählt, wie oft etwas hinzugefügt und heruntergenommen wurde.