Aufgabe 3 Nebenläufigkeit **(20 Punkte)**

Entwickeln Sie ein System zur Verwaltung eines Parkhauses. Das Parkhaus hat insgesamt 200 Plätze und ein Schrankensystem (jeweils eine Schranke für die Ein- und Ausfahrt), um sicherzustellen, dass nie mehr Autos in das Parkhaus fahren als Plätze vorhanden sind. Gehen Sie in allen Aufgaben davon aus, dass die zu entwickelnden Prozesse bei einem leeren Parkhaus starten.

Lösen Sie die folgenden Aufgaben unter Verwendung von Pseudocode und den in der Vorlesung behandelten Prozesssynchronisations-Mechanismen.

a) Realisieren Sie die beiden Schrankenprozesse für die Ein- und Ausfahrt, wobei die wesentlichen Elemente das Warten auf einen Autofahrer, das Öffnen der Schranke (zur Vereinfachung wird angenommen, dass die Schranke selbstständig schließt und der Autofahrer immer rechtzeitig den Schrankenbereich durchquert), sowie die Verwaltung der Parkhausplätze sind. Für die ersten beiden Schritte können Sie die (in dieser Aufgabe nicht zu implementierenden) Funktionsaufrufe warteAufFahrer() und öffneSchranke() benutzen.

b) Um Autofahrern zu signalisieren, dass das Parkhaus bereits voll ist, wird an der Straße eine Anzeige angebracht, die anzeigt ob das Parkhaus noch frei oder bereits besetzt ist. Realisieren Sie einen entsprechenden Prozess für die Anzeige und modifizieren Sie gegebenenfalls die in Aufgabe a) realisierten Prozesse. Verwenden Sie die Funktionsaufrufe zeigFrei() und zeigBesetzt() zum Stellen der Anzeige. Achten Sie dabei auf eine effiziente Implementierung und verzichten Sie auf globale Variablen.

c) Um den Komfort der Anzeige zu erhöhen, soll nun die Anzeige anstelle von frei und besetzt die konkrete Anzahl der freien Plätze angezeigt werden. Verwenden Sie den Funktionsaufruf zeig(i), um die Zahl i am Display anzuzeigen. Achten Sie dabei auf eine effiziente Implementierung und verzichten Sie auf globale Variablen.