VO 182.711 4. Juni 2014

Prüfung Betriebssysteme

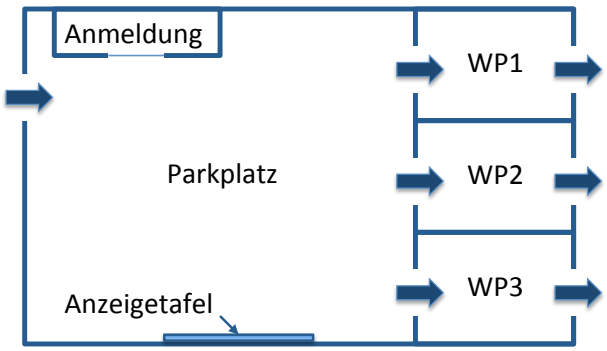
KNr. MNr. Zuname, Vorname

Ges.)(100)	1.)(35)	2.)(20)	3.)(45)	Zusatzblätter:

Bitte verwenden Sie nur dokumentenechtes Schreibmaterial!

1 Synchronisation mit Semphoren (35)

Die Abläufe in einer Autoreparaturwerkstätte werden auf einem Computersystem durch eine Menge von Prozessen simuliert. Für die Simulation sind folgende Objekte von Bedeutung (siehe Skizze):



- Beim Anmeldeschalter (Anmeldung), den ein Auto beim Einfahren in den Werkstättenbereich passiert, melden Fahrer ihr Fahrzeug zur Reparatur an. Bei der Anmeldung bekommt der Fahrer eine Identifikationsnummer (ID), die er für die spätere Einfahrt auf einen Werkplatz benötigt. Beim Anmeldeschalter kann sich zu jedem Zeitpunkt maximal ein Fahrzeug aufhalten.
- Auf dem Parkplatz (gesamtes Areal zwischen linker Einfahrt und den Einfahrten zu WP1, WP2 und WP3) können sich maximal acht Fahrzeuge befinden. Fahrzeuge warten auf dem Parkplatz auf das Freiwerden eines Werkplatzes.
- Es gibt drei Werkplätze (WP1, WP2 und WP3), an denen Autos begutachtet bzw. repariert werden. Auf jedem dieser Plätze hat maximal ein Auto Platz.
- Auf dem Parkplatz befindet sich eine Anzeigetafel, die für jeden der drei Werkplätze die ID desjenigen Fahrzeugs anzeigt, das sich als nächstes für das Einfahren auf den Werkplatz bereithalten soll. Diese Reihenfolge kann von der Anmeldereihenfolge der Fahrzeuge abweichen. Die Aktualiserung der Anzeigetafel erfolgt mit folgendem Codestück:

```
P(display)
update_display()
V(display)
```

Schreiben Sie einen Prozess für einen Fahrer, der sein Fahrzeug unter Einhaltung der obigen Einschränkungen in der Werkstatt reparieren lässt. Es sollen beliebig viele solche Prozesse gestartet werden können. Verwenden Sie *Semaphore* zur Synchronisation der Prozesse und vermeiden Sie unnötige Einschränkungen der Parallelität. Ein Fahrer führt folgende Schritte bei einem Werkstattbesuch durch:

- Wenn Platz ist, meldet sich der Fahrer mit der Funktion `register()` beim Anmeldeschalter an und bekommt eine ID als Ergebniswert des Funktionsaufrufs.
- Der Fahrer parkt sein Auto (`park_car()`) auf dem Parkplatz und wartet auf die Zuweisung eines Werkplatzes: dabei ruft er abwechselnd die Funktionen `wait()` und `check_display()` auf, die das Warten bzw. das Ablesen der Anzeigetafel realisieren. An `check_display()` übergibt der Prozess die an der Anmeldung zugewiesene ID. Der Returnwert der Funktion gibt an, ob und für welchen Werkplatz die Nummer ID auf der Anzeigetafel steht – Returnwert 0: ID steht nicht auf der Anzeigetafel, Returnwert $x \in \{1, 2, 3\}$: Der Fahrer soll sich mit seinem Fahrzeug für das Einfahren auf Werkplatz WPx bereitmachen.
  Achten Sie darauf, dass Ihre Lösung das gleichzeitige Ablesen der Tafel durch mehrere Fahrer erlaubt.
- Der Fahrer wartet bis der ihm zugewiesene Werkplatz frei wird, fährt dann auf den Werkplatz (Aufruf von `enter_wp()` mit der Werkplatznummer als Parameter), lässt die Reparatur durchführen (Aufruf von `repair_car()`), und verlässt den Werkplatz durch die Ausfahrt, auf der Skizze rechts (Aufruf von `exit_wp()`, mit der Werkplatznummer als Parameter).

Lösen Sie die Synchronisationsaufgabe mit *Semaphoren* und geben Sie Initialisierungen für die Semaphore an. Achten Sie bei der Implementierung des Fahrerprozesses darauf, dass die Lösung die Ausführung und Synchronisation einer "beliebigen" Anzahl von Fahrern/Fahrzeugen erlaubt.

Code für Prozesse und Initialisierungen

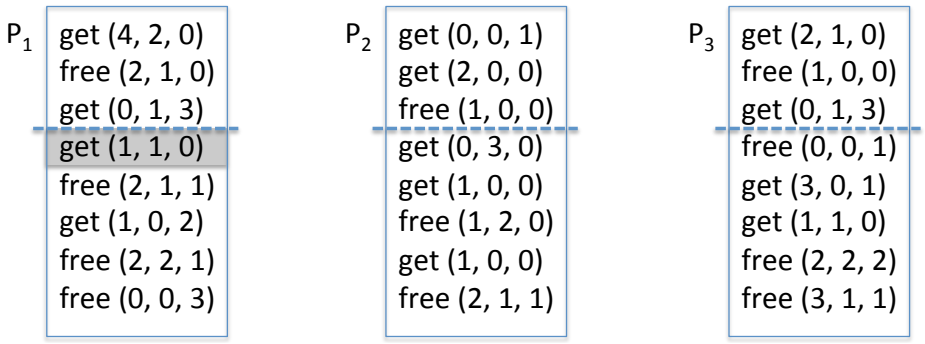
`Initialisierungen:`

Prozess Fahrer:

2 Deadlock Avoidance – Banker's Algorithm (20)

In einem Computersystem gibt es drei Prozesse (P_1, P_2, P_3) und drei Arten von Ressourcen (R_1, R_2, R_3). Der *Resource Vector* R=(7, 8, 7) beschreibt die Anzahl der vorhandenen Ressourcen. Prozesse verwenden die Operationen *get*(r_1, r_2, r_3) bzw. *free*(r_1, r_2, r_3), um r_i Ressourcen der Ressourcenart R_i ($i = 1 \ldots 3$) anzufordern bzw. freizugeben.

Die folgende Abbildung zeigt für jeden der drei Prozesse die Folge von *get* und *free*-Operationen, die bei jeder Prozessabarbeitung durchgeführt werden.



Nehmen Sie an, dass die drei ersten Operationen jedes Prozesses abgearbeitet wurden (d.h., die Abarbeitung jedes Prozesses befindet sich an der strichlierten Linie). Als nächster Schritt soll die vierte Operation von P_1 (grau hinterlegte Operation) durchgeführt werden.

Verwenden Sie den Banker's Algorithmus, um festzustellen, ob die vierte Operation von P_1 durchgeführt werden soll. Geben Sie alle erforderlichen Vektoren und Matrizen an und führen Sie den Banker's Algorithmus schrittweise durch, d.h. geben Sie für jeden Schritt die Werte der Elemente aller relevanten Matrizen und Vektoren an.

3 Fragen zu Betriebssystemen (45)

Was versteht man unter einem *Microkernel*? Welche Services stellt ein Microkernel zur Verfügung? Welche Vor- bzw. Nachteile ergeben sich bei der Verwendung eines Microkernel-Betriebssystems? (5)

Skizzieren Sie die Folge der Schritte, die bei der Abarbeitung eines *Synchronen I/O Requests* ablaufen. (5)

Erklären Sie die Begriffe *Process Switch* und *Mode Switch*, sowie die Beziehung, in der diese beiden Konzepte stehen. Zählen Sie weiters die drei Klassen von Ereignissen auf, die einen Mode Switch nach sich ziehen. (4)

Worin liegt der grundlegende Unterschied zwischen *Prozessen* und *Threads*? Welcher Vorteil ergibt sich aus der Einführung von Threads für den Benutzer und worauf muss der Benutzer achten? (4)

Beschreiben Sie die folgenden Strategien für das CPU Scheduling und vergleichen Sie deren Eigenschaften: *First Come First Serve*, *Round Robin* und *Virtual Round Robin*. (4)

Was versteht man unter *Virtual Memory Management*? Welche Vorteile bietet es? (5)

Welche Möglichkeiten kennen Sie, um in einem Paging-System Speicherschutz zu realisieren? (3)

Was versteht man unter *Buffering*? Welche Vorteile bietet es, wo liegen seine Grenzen und worauf hat man bei der Verwendung von Puffern bei der Betriebssystemimplementierung zu achten? (5)

Nennen Sie Design Prinzipien für die Konstruktion von sicheren Systemen. Geben Sie für jede Regel ein Beispiel an. (4)

Was ist *Swapping*? Wann wird es angewandt? (2)

Nennen Sie die vier Layer eines TCP/IP Stacks und beschreiben Sie kurz deren Aufgaben. Geben Sie evtl. Beispiele an. (4)