VO 182.711 5. Juni 2013

Prüfung Betriebssysteme

KNr. | MNr. | Zuname, Vorname

Ges.)(100)	1.)(30)	2.)(25)	3.)(45)	Zusatzblätter:

Bitte verwenden Sie nur dokumentenechtes Schreibmaterial!

1 Synchronisation mit Semaphoren (30)

Bei einem Laufwettbewerb treten jeweils drei Läufer gegeneinander auf einem Hindernisparcour an. Da der Wettlauf einige Besonderheiten aufweist, soll er zunächst auf einem Computersystem simuliert werden, wobei die Läufer durch drei Prozesse realisiert werden. Ein Wettlauf läuft wie folgt ab.

- Die Läufer wärmen sich auf (*warm_up()*) und begeben sich dann in die Startzone für den Lauf (*enter_start_zone()*).
- Sobald sich alle drei Läufer in der Startzone befinden, beginnt der Wettlauf. Die Läufer durchlaufen den ersten Abschnitt des Parcours (*section1()*).
- Nach dem ersten Abschnitt des Parcours müssen die Läufer eine Hängebrücke passieren (*pass_bridge()*). Aus Sicherheitsgründen dürfen die Läufer nur mit einem zeitlichen Mindestabstand von fünf Sekunden auf die Brücke laufen (d.h., hier müssen Läufer eventuell warten, wenn sie knapp hinter einem schnelleren Läufer bei der Brücke ankommen).
- Zur Einhaltung des Zeitabstands beim Betreten der Brücke synchronisieren sich die Läufer mit dem Prozess *Clock*, dessen Kernstück wie folgt implementiert ist.

```
/* Code Process Clock */
    ...
while(1){
    P(entered_bridge);   /* wait until a runner enters the bridge */
    for (i=0; i<5; i++){ /* delay next runner for 5*1 seconds  */
        delay(1);        /* wait for one second */
        V(second);       /* signal that one second has passed */
    }
}
```

- Nach dem Passieren der Brücke durchlaufen die Läufer den zweiten Abschnitt des Parcours (*section2()*).
- Der Wettlauf endet durch das Passieren des Zieleinlaufs (*finish()*), wobei immer nur ein Läufer gleichzeitig ins Ziel laufen kann.

Vervollständigen Sie die Initialisierungen und die gegebenen Prozesse zur Simulation der drei Läufer, die sich wie beschrieben verhalten. Die Läufer-Prozesse sollen die im Text genannten Funktionen aufrufen, um die einzelnen Phasen eines Laufes zu simulieren. Verwenden Sie ausschließlich Semaphore, um die Prozesse zu synchronisieren und die angegebenen Bedingungen und Ereignisfolgen sicherzustellen. Die Verwendung von anderen Synchronisationskonstrukten und globalen Variablen ist nicht erlaubt.

Initialisierungen

```
init(entered_bridge, 0);  init(second, 5);
```



```
/** Code Laeufer1 **/
```



```
/** Code Laeufer2 **/
```



```
/** Code Laeufer3 **/
```

2 Page Replacement (25)

Gebeben ist ein Arbeitsspeicher mit vier Frames, dessen Seiten mit unterschiedlichen Ersetzungsstrategien (OPT, FIFO, und LRU) ersetzt werden sollen. Die Seitenzugriffsfolge ist für alle Algorithmen gleich. Sie ist jeweils in der Kopfzeile der Tabelle gegeben. Geben Sie in den Spalten der Tabellen die Speicherinhalte für jeden Frame nach dem in der Kopfzeile angegebenen Seitenzugriff an und kennzeichnen Sie in der letzten, mit *PF* markierten Zeile das Auftreten von Page Faults. Der Arbeitsspeicher ist am Beginn leer und wird zunächst mit Frame 0 beginnend befüllt.

OPT-Strategie:

	A	B	C	D	E	D	F	C	B	E	G	G	H	E	C	E
0																
1																
2																
3																
PF																

FIFO-Strategie:

	A	B	C	D	E	D	F	C	B	E	G	G	H	E	C	E
0																
1																
2																
3																
PF																

LRU-Strategie:

	A	B	C	D	E	D	F	C	B	E	G	G	H	E	C	E
0																
1																
2																
3																
PF																

Vergleichen Sie die Anzahl der bei den Seitenersetzungsstrategien beobachteten Page Faults. Entspricht das Ergebnis Ihren Erwartungen? Warum bzw. warum nicht?

3 Fragen zu Betriebssystemen (45)

Was versteht man unter einem *Process Control Block*? Beschreiben Sie, aus welchen Teilen der PCB besteht und welche Informationen in diesen Teilen jeweils verwaltet werden. (6)

Was versteht man unter einem *Monitor* zur Prozesssynchronisation? Nennen Sie die wichtigsten Komponenten und Eigenschaften des Monitors. (4)

Beschreiben Sie die folgenden Strategien für das CPU Scheduling und vergleichen Sie deren Eigenschaften: *FCFS*, *Round Robin*, *SPN* und *SRT*. (4)

Was versteht man unter der *Working-Set Strategie*? Beschreiben Sie deren Funktionsweise und erklären Sie, wie diese Strategie zur Optimierung eines Paging-Systems eingesetzt werden kann. (5)

Welche Informationen werden in den Einträgen einer *Page Table* gespeichert? Nennen und beschreiben Sie diese kurz. (3)

Beschreiben Sie den Aufbau und Verwendung einer *Multilevel Page Table* (eventuell mit Skizze). Warum werden Multilevel Page Tables verwendet? (3)

Mit welcher logischen Struktur des I/O Systems versucht man bei der Realisierung von I/O Funktionen sowohl ein einheitliches Programmierinterface, als auch eine möglichst gerätespezifische Ansteuerung zu erreichen? (4)

Beschreiben Sie das Prinzip einer Sicherheitsattacke durch Stack/Buffer Overflow. Wodurch kann man sich bei der Implementierung eines Betriebssystems vor einen solchen Angriff schützen? (4)

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

Wodurch unterscheidet sich ein *Network Operating System* von einem *Distributed Operating System*? (4)

Was versteht man in einem Distributed File System unter *Location Transparency* bzw. *Location Independence*. Erklären Sie die beiden Begriffe. (2)

Nennen Sie drei Möglichkeiten, wie *Distributed Mutual Exclusion* realisiert werden kann. Charakterisieren Sie jede dieser Möglichkeiten kurz. (4)