Punkte:	7	4	13	8	4	7

Name, Vorname: _____ **Immatr.-Nr.: _____**

Prüfungsklausur Betriebssysteme
am 22. Februar 2002

Aufgaben 1 bis 4: Bearbeitungszeit 60 Minuten

Aufgaben 1 bis 6: Bearbeitungszeit 90 Minuten

1. a) In einem Multiprozessorsystem gebe es eine gemeinsam genutzte Variable

```
int z = 7;
```

und ein Programm

```
{ z = z+1; }
```

das von zwei Prozessen A, B einmalig abgearbeitet wird. Warum hat nach Beendigung von A und B die Variable `z` nicht notwendig den Wert 9? Welche Werte kann sie haben?

b) Modifizieren Sie das Programm so, daß der Endwert 9 garantiert ist! (Die Anweisung `z = 9` ist dabei ausgeschlossen.)

c) Das folgende Programm initialisiert die Variable `i` mit 0, öffnet durch `fopen` eine Datei namens `i.txt` mit Schreibrecht und erzeugt danach einen weiteren Prozeß. Jeder der nunmehr zwei Prozesse erhöht die Variable `i` um 1 und schreibt sie mit `fprintf` als Dezimalzahl in die geöffnete Datei. Welchen Inhalt hat die Datei nach Ende beider Prozesse (unter der Annahme, daß es keine Fehler während der Programmausführung gibt)? (Begründung!)

```
#include <stdio.h>
#include <unistd.h>
int main(void)
{
  int i = 0;
  FILE *f = fopen("i.txt", "w");
  if (f) {
    if (fork() >= 0) {
      i = i+1;
      fprintf(f, "%d", i);
    }
    fclose(f);
  }
  return 0;
}
```

2. Ein Prozeßsystem bestehe aus vier Prozessen P_1,..., P_4 und vier Betriebsmitteln A, B, C, D. Dabei gebe es von A, B und D je ein Exemplar und von C mehrere Exemplare. Das Betriebssystem teilt ein freies Betriebsmittel-Exemplar nur *einem* Prozeß zu. Außerdem erfolgt die Zuteilung eines Exemplars nur, nachdem es ein Prozeß freigegeben hat (abgesehen vom Anfangszustand). Ein Prozeß gibt erst dann seine Betriebsmittel-Exemplare frei, wenn er alle angeforderten Exemplare auch erhalten hat; sofern möglich, erhält er alle angeforderten Exemplare auf einmal, sonst gar keine. Betrachtet werde folgender Zustand:
   - P_1 besitzt nichts
   - P_2 besitzt 1 Exemplar von C
   - P_3 besitzt je 1 Exemplar von A und B
   - P_4 besitzt je 1 Exemplar von C und D.

   Jeder der Prozesse P_1, P_2 und P_4 fordert nun je ein Exemplar von A und C an (in einer gemeinsamen Anforderung), P_3 stellt keine weiteren Forderungen. Welches ist die Mindestanzahl n von Exemplaren des Betriebsmittels C, so daß keine Verklemmung auftritt? Begründen Sie, daß es bei $n - 1$ Exemplaren zwangsläufig zu einer Verklemmung kommt; verwenden Sie dazu auch einen Prozeß-Betriebsmittel-Graphen!

3. In einem Echtzeitsystem sind periodische Tasks auf der Basis von statischen Prioritäten so einzuplanen, daß sie in jeder Periode erfolgreich beendet werden (d.h., das Periodenende ist die Deadline). Für vier Tasks T_1,...,T_4 liegen folgende Angaben vor (t_i: Bearbeitungszeit, p_i: Periodenlänge):

   T_1: $t_1 = 2$, $p_1 = 12$; T_2: $t_2 = 2$, $p_2 = 48$; T_3: $t_3 = 1$, $p_3 = 6$; T_4: $t_4 = 1$, $p_4 = 4$.

   a) Begründen Sie, daß diese Tasks mittels RMS eingeplant werden können! Wie lautet die Prioritätszuordnung?

   b) Ist dies auch möglich, wenn eine weitere Task T_5: $t_5 = 3$, $p_5 = 8$ einzuplanen ist? (Begründung!)

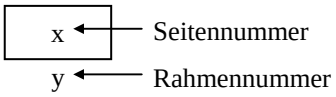
   c) Wenn nein, ist es dann auf andere Weise möglich, den Tasks Prioritäten statisch so zuzuordnen, daß alle Tasks stets erfolgreich sind? Wenn ja, wie lautet die Prioritätszuordnung; wenn nein, warum nicht?

   d) Sollten sich die fünf Tasks nicht mittels statischer Prioritäten einplanen lassen, können sie dann überhaupt eingeplant, d.h. irgendwie so abgearbeitet werden, daß sie stets ihre Deadline einhalten? Wenn ja, wie und warum?

   Der Scheduling-Overhead werde in allen Fällen vernachlässigt.

4. Gegeben sei ein System mit virtuellem Speicher, das folgende Eigenschaften hat:
   - 32-Bit-Adressen im virtuellen und physischen Speicher
   - 64MByte physischer Speicher
   - Seitengröße 4Kbyte (2^{12} Byte)
   - einstufige Seitentabellen.

   In diesem System laufe ein Prozeß, dessen Adreßraum die in Abb. 1 dargestellte Struktur hat (die Adressen kennzeichnen jeweils den Beginn der entsprechenden Bereiche); der Textbereich ist nur lesbar, die anderen Bereiche (Data, BSS, Stack) auch schreibbar. Abb.2 beschreibt den aktuellen Zustand der Seitentabelle; Seiten mit COW-Bit werden noch von anderen Prozessen benutzt. Das Betriebssystem verwendet zur Seitenersetzung den Clock-Algorithmus; die aktuelle Liste ist in Abb. 3 dargestellt, Bedeutung:



   Gegenwärtig ist nur noch Kachel (Rahmen) 123 frei verfügbar. Sämtliche Zahlenangaben sind hexadezimal.

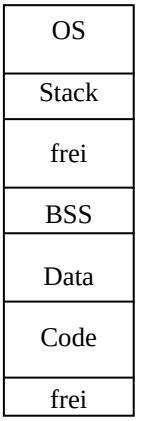


Abb. 1. Adreßraumstruktur

…						
E	53		u	ro		
F	200		u	ro	p	
10	340		u	ro	p	
11	38		u	ro	p	
12	68		u	ro		
13	88		u	rw		
14	321	cow	u	ro	p	
15	657		u	rw		
…						
BFFFF	42		u	rw	p	
…						

Abb. 2. Seitentabelle (Ausschnitt)

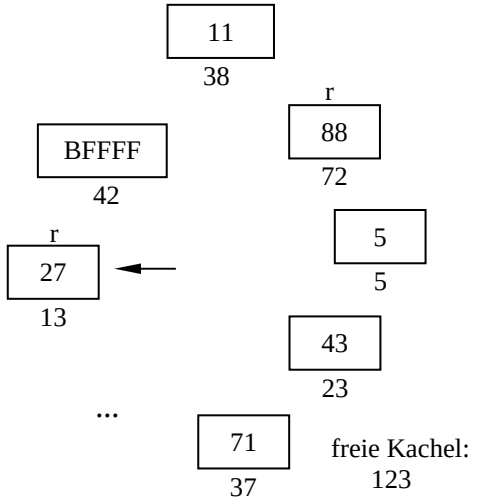


Abb. 3. Clock-Liste (Ausschnitt)

Vollziehen Sie die Reaktionen von Betriebssystem und Hardware auf die in untenstehender Tabelle angegebene Folge von Speicherzugriffen des Prozesses im User-Mode nach. Bestimmen Sie jeweils die physische Adresse, tragen Sie die ggf. erforderlichen Änderungen in Abb. 2 und 3 ein. Ist ein Zugriff illegal, so begründen Sie dies in der Tabelle und ignorieren Sie den Zugriff für das weitere Vorgehen (normalerweise würde der Prozeß abgebrochen).

virtuelle Adresse	Modus	physische Adresse bzw. Begründung
BFFFFABE	read	
1038A	write	
12034	read	
13304	write	
11200	read	
14198	write	
E278	read	

5. In einem verteilten System gebe es einen Dateiserver und mehrere Clients, die nach folgendem Prinzip arbeiten:
   - Dateien werden blockweise übertragen;
   - Dateiblöcke werden beim Client zum Lesen im Hauptspeicher als Cache zwischengespeichert;
   - Schreiboperationen eines Clients erfolgen nach dem write-through-Prinzip direkt beim Server.

   Ein Client möchte nun eine komplette Datei vom Server holen, von der einige Blöcke in seinem Cache stehen, andere nicht. Erklären Sie das wichtigste Problem, das dabei auftreten kann, und geben Sie eine Lösungsmöglichkeit an!

6. Betrachtet werde folgendes Problem. Für ein System paralleler Prozesse existiere ein gemeinsam genutzter Puffer (der als Ganzes betrachtet wird). Dieser Puffer wird von einem Prozeß S beschrieben und von prinzipiell beliebig vielen Prozessen L gelesen, allerdings ist ein gleichzeitiges Lesen nur durch maximal fünf Prozesse zugelassen; außerdem ist kein gleichzeitiges Lesen und Schreiben möglich.

   Dazu sei folgender Lösungsversuch gegeben:

```
                    int z = 0;
                    sem_t *S = new_sem(1);
                    sem_t *L = new_sem(5);

Leserprozeß: while(1)  {                    Schreiberprozeß:    while (1)  {
                 z = z+1;                                          P(S);
                 if (z == 1)                                       schreiben();
                   P(S);                                           V(S);
                 P(L);                                         }
                 lesen();
                 V(L);
                 z = z-1;
                 if (z == 0)
                   V(S);
             }
```

   a) Welchem Zweck dienen die drei Variablen z, *S und *L?

   b) Die Lösung ist nicht korrekt. Worin liegt die Ursache?

   c) Geben Sie eine korrekte Lösung an, indem Sie die vorliegende Lösung geeignet modifizieren!