Aufgabe 1: (20 Punkte)

Bei den Multiple-Choice-Fragen ist jeweils nur **eine** richtige Antwort eindeutig anzukreuzen. Auf die richtige Antwort gibt es die angegebene Punktzahl.

Wollen Sie eine Multiple-Choice-Antwort korrigieren, kreisen sie bitte die falsche Antwort ein und kreuzen die richtige an.

Lesen Sie die Frage genau, bevor Sie antworten.

a) Prozessoren kennen oft einen Benutzermodus und einen privilegierten Betriebsmodus. Welche Aussage ist **falsch**? 2 Punkte

- ❏ Die Konfiguration des Prozessors kann nur im privilegierten Modus verändert werden.
- ❏ Im Benutzermodus ist der Befehlssatz des Prozessors eingeschränkt.
- ❏ Wird zwischen zwei Prozessen des gleichen Benutzers umgeschaltet, so ist kein Wechsel in den privilegierten Modus nötig.
- ❏ Spezielle Befehle zur Ansteuerung externer Geräte (z.B. Festplatte) stehen nur im privilegierten Modus zur Verfügung.

b) Welche Aussage bezüglich Seitenersetzungsstrategien ist richtig? 2 Punkte

- ❏ FIFO-Anomalie bedeutet, dass ein größerer Hauptspeicher (mit mehr Speicherkacheln) immer zu einem schlechteren Verhalten führt.
- ❏ In den meisten Betriebssystemen wird die B_0-Strategie verwendet, da sie die besten Ergebnisse liefert.
- ❏ Die LRU-Strategie (Least Recently Used) versucht der B_0-Strategie möglichst nahe zukommen, indem immer die Seite ersetzt wird, welche am längsten unbenutzt war.
- ❏ Die FIFO-Anomalie tritt nur bei der FIFO-Ersetzungsstrategie auf.

c) Welche Aussage über einem UNIX-Systemaufruf ist **falsch**? 2 Punkte

- ❏ Bei Systemaufrufen wird mit Hilfe eines *User Interrupts* in den privilegierten Modus des Prozessors gewechselt.
- ❏ Parameter werden nach einer festen Konvention an das System übergeben.
- ❏ Alle Register, insbesondere der Programmzähler, werden gesichert.
- ❏ Nach dem Umschalten in den privilegierten Prozessormodus wird eine vom Benutzer festgelegte Befehlsfolge ausgeführt.

d) Welche Daten werden typischerweise nicht im UNIX-Prozesskontrollblock gespeichert? 1 Punkt

- ❏ UID des Eigentümers
- ❏ offene Dateien
- ❏ aktuelles Arbeitsverzeichnis
- ❏ Homeverzeichnis des Eigentümers

e) Sie haben den Algorithmus von Peterson zur Koordinierung eines kritischen Abschnittes kennengelernt. Welche Aussage ist **falsch**? 2 Punkte

- ❏ Der Algorithmus von Perterson benötigt keine Semaphoren.
- ❏ Wenn der kritische Abschnitt besetzt ist, so wartet der Algorithmus aktiv.
- ❏ Zur Sicherung des kritischen Abschnittes werden spezielle Maschinenbefehle verwendet.
- ❏ Der Algorithmus von Peterson benötigt nur drei Variablen.

f) Welche Aussage ist **falsch**? 2 Punkte

- ❏ Die Prüfoperation beim Hochfahren eines UNIX-Systems kann auf einem typischen UNIX-Dateisystem (z.B. UFS) inkonsistente Zustände erkennen und beseitigen. Dabei gehen unter Umständen Daten verloren.
- ❏ Journalled-Dateisysteme benötigen keine zusätzliche Operation beim Hochfahren des Systems, da ihre Daten immer konsistent sind.
- ❏ Das Prüfen der Konsistenz eines Dateisystems beim Hochfahren kann unterbleiben, wenn das System vor dem Herunterfahren die Konsistenz erkannt und im Dateisystem markiert hat. Vor der Veränderung eines so markierten Dateisystems muss die Marke entfernt werden.
- ❏ Bei Journalled-Dateisystemen wird die Konsistenz durch ein Transaktionskonzept mit Protokollierung der Änderungen sichergestellt.

g) Welche der folgenden Aussagen **gehört nicht** zu den notwendigen Bedingungen für eine Verklemmung? 3 Punkte

- ❏ Mindestens ein Betriebsmittel kann nur exklusiv belegt werden.
- ❏ Betriebsmittel können nicht entzogen werden.
- ❏ Es muss einen Prozess geben, der mindestens zwei Betriebsmittel gleichzeitig benötigt.
- ❏ Es muss eine totale Ordnung auf den Betriebsmitteln definiert sein.

h) Welche Aussage bezüglich des Systemaufrufs `select()` ist richtig? 2 Punkte

- ❏ Mit `select()` kann eine Datei ausgewählt werden, welche anschließend geöffnet wird.
- ❏ `select()` kann bei TCP-Kommunikation über Sockets verwendet werden, um eine von mehreren anstehenden Verbindungsanforderungen zu akzeptieren.
- ❏ Mittels `select()` kann der Zustand von File-Deskriptoren überwacht werden.
- ❏ Die primäre Aufgabe von `select()` ist es eine bestimmte Zeitspanne zu warten.

i) Sie kennen den Translation-Look-Aside-Buffer (TLB). Welche Aussage ist richtig? 2 Punkte

- ❏ Der TLB verkürzt die Zugriffszeit auf den physikalischen Speicher, da ein Teil des Speichers in einem sehr schnellen Pufferspeicher vorgehalten wird.
- ❏ Der TLB puffert die Ergebnisse der Abbildung von physikalische auf logische Adressen, sodass eine erneute Anfrage sofort beantwortet werden kann.
- ❏ Verändert sich die Speicherabbildung von logische auf physikalische Adressen aufgrund einer Adressraumumschaltung, so bleiben die Daten im TLB gültig.
- ❏ Der TLB ist eine schnelle Umsetzeinheit der MMU, die logische in physikalische Adressen umsetzt.

j) Betrachten Sie folgende Aussagen zur Speicherung der Daten einer Datei auf Festplatte. Welche Aussage ist **falsch**? 2 Punkte

- ❏ Eine kontinuierliche Speicherung der Daten in aufeinanderfolgenden Blöcken erhöht die Lese- und Schreibleistung gegenüber verstreuter Speicherung.
- ❏ Bei kontinuierlicher Speicherung kann das Auffinden eines genügend großen zusammenhängenden Speicherbereiches schwierig sein.
- ❏ Kontinuierliche Speicherung ist gänzlich unmöglich, falls Dateien dynamisch erweiterbar sein sollen.
- ❏ Bei kontinuierlicher Speicherung kann die Ortsinformation lediglich aus der Nummer des ersten Plattenblocks und der Dateilänge bestehen.

Aufgabe 2: (40 Punkte)

2 Punkte

a) Schreiben Sie ein Programm **`batchd`** (Batch-Job-Daemon), das Aufträge, die von einem Auftraggeber in ein spool-Directory eingetragen werden, ausführt.

Die Auftraggeber legen in dem spool-Directory Dateien mit Namen cf*xxx* (*xxx* steht für eine beliebige Zeichenfolge) an, die den Namen eines auszuführenden Kommandos enthalten (lediglich ein Kommando ohne Parameter). Anschliessend informiert ein Auftraggeber mit dem Signal SIGUSR1 den batchd darüber, dass neue Aufträge vorliegen.

Das **`batchd`**-Programm erhält beim Start als Argument den Namen des spool-Directories. Es läuft danach in einer Endlosschleife.

- Im Rahmen eines Durchlaufs sucht batchd in dem spool-Directory nach cf*xxx*-Dateien. Andere Dateien werden ignoriert.
- Wird eine cf*xxx*-Datei gefunden, wird das darin angegebene Kommando ausgeführt (das Kommando soll über die PATH-Variable gesucht werden). Nach Beendigung der Kommandoausführung wird die Datei gelöscht.
- Der Programmteil der sich um die Kommandoausführung kümmert ist in einer eigenen Funktion namens **`execute_job`** zu programmieren. Die Funktion erhält als Parameter den Namen der cf*xxx*-Datei.
- Werden keine weiteren cf*xxx*-Dateien mehr gefunden, legt sich der **`batchd`** schlafen bis ein Signal SIGUSR1 eintrifft. Danach beginnt er einen neuen Schleifendurchlauf.
- Während eines Schleifendurchlaufs wird das Eintreffen von SIGUSR1 blockiert.
- Kann eine cf*xxx*-Datei nicht geöffnet werden, wird der Fehler gemeldet, der batchd aber nicht abgebrochen.
- Gehen Sie davon aus, dass alle benutzten Datei- und Directorynamen jeweils nie länger als 1024 Zeichen sein werden.

Auf den folgenden Seiten finden Sie ein Gerüst für das beschriebene Programm. In den Kommentaren sind nur die wesentlichen Aufgaben der einzelnen, zu ergänzenden Programmteile beschrieben, um Ihnen eine gewisse Leitlinie zu geben. Es ist überall sehr großzügig Platz gelassen, damit Sie auch weitere notwendige Programmanweisungen entsprechend Ihrer Programmierung einfügen können.

Einige wichtige Manual-Seiten liegen bei - Sie können aber nicht davon ausgehen,

dass Sie ausschließlich nur diese Funktionen benötigen.

```
/* includes */
#include <stdio.h>
#include <dirent.h>
#include <signal.h>
#include <errno.h>


/* Funktionsdeklarationen, globale Variablen */

.....

.....

.....

.....

.....

/* Funktion main */

.....

.....
{
  /* lokale Variablen und was man sonst am Anfang so braucht */

    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
    .....
```

A:


```
/* Signalbehandlung einrichten */

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
```

S:

```
/* Hauptschleife */
```

```
} /* Ende Funktion main */
```

D:


```
/* Signalhandler für SIGUSR1 */
```

S: 1

```
/* Funktion execute_job */
```

```
{
  /* lokale Variablen etc. */
```

```
  /* Datei öffnen, Kommando einlesen*/
```

P:
S:1

```
/* Prozess zur Kommandoausführung erzeugen */
.....
.....
.....
.....

/* Fehler bei Prozesserzeugung */
.....
.....
.....
.....
.....

/* Sohnprozess */
.....
.....
.....
.....
.....
.....
.....
.....
.....

/* Vaterprozess */
.....
.....
.....
.....
.....
.....
.....

} /* Ende Funktion execute_job */
```

P:

b) Schreiben Sie ein Makefile zum Erzeugen des batchd-Programms.
Ein Aufruf von

```
make batchd
```

soll das Programm erzeugen, ein Aufruf von

```
make clean
```

soll das **batchd**-Programm und evtl. erzeugte **.o**-Dateien entfernen.

.....
.....
.....
.....
.....
.....
.....
.....
.....

M: