Aufgabe 1: Ankreuzfragen (7 Punkte)

1) Einfachauswahlfragen (4 Punkte)

Bei den Einfachauswahlfragen in dieser Aufgabe ist jeweils nur **eine** richtige Antwort eindeutig anzukreuzen. Auf die richtige Antwort gibt es die angegebene Punktzahl.

Wollen Sie eine Antwort korrigieren, streichen Sie bitte die falsche Antwort mit drei waagrechten Strichen durch (☒) und kreuzen die richtige an.

Lesen Sie die Frage genau, bevor Sie antworten.

a) Welche Aussage über Variablen und Funktionen in C-Programmen ist richtig? 2 Punkte

- ☐ Wird dem Parameter einer Funktion innerhalb der Funktion ein neuer Wert zugewiesen, so ändert sich auch der Wert der Variablen, die von der aufrufenden Funktion als Parameter übergeben wurde.
- ☐ Es ist nicht möglich, Zeiger als Parameter an Funktionen zu übergeben.
- ☐ Eine Funktion, die mit dem Schlüsselwort static definiert wird, kann nur innerhalb des Moduls verwendet werden, in dem sie definiert wurde, nicht jedoch aus einem anderen Modul heraus.
- ☐ Lokale automatic-Variablen, die auf dem Stack angelegt werden, werden immer mit dem Wert 0 initialisiert.

b) Was muss geschehen, damit sich ein Linux-Prozess im Zustand Zombie befindet? 2 Punkte

- ☐ Der Elternprozess muss terminieren, aber der init-Prozess (PID 1) darf den Kindprozess noch nicht adoptieren.
- ☐ Der Prozess muss terminieren, aber der Exitstatus darf noch nicht abgefragt worden sein.
- ☐ Der Prozess muss blockieren und diesen Zustand nie wieder verlassen.
- ☐ Die Terminierung des Prozesses muss fehlschlagen.

2) Mehrfachauswahlfragen (3 Punkte)

Bei den Mehrfachauswahlfragen in dieser Aufgabe sind jeweils m Aussagen angegeben, davon sind n ($0 \leq n \leq m$) Aussagen richtig. Kreuzen Sie alle richtigen Aussagen an.
Jede korrekte Antwort in einer Teilaufgabe gibt einen Punkt, jede falsche Antwort einen Minuspunkt. Eine Teilaufgabe wird minimal mit 0 Punkten gewertet, d. h. falsche Antworten wirken sich nicht auf andere Teilaufgaben aus.

Wollen Sie eine falsch angekreuzte Antwort korrigieren, streichen Sie bitte das Kreuz mit drei waagrechten Strichen durch (⊗).

Lesen Sie die Frage genau, bevor Sie antworten.

a) Man unterscheidet zwischen Traps und Interrupts. Welche der folgenden Aussagen ist richtig?

3 Punkte

- ○ Die CPU sichert bei einem Interrupt einen Teil des Prozessorzustands.
- ○ Weil das Betriebssystem nicht vorhersagen kann, wann ein Prozess einen Systemaufruf tätigt, sind Systemaufrufe in die Kategorie Interrupt einzuordnen.
- ○ Der Zugriff auf eine logische Adresse kann zu einem Trap führen.
- ○ Ein Programm darf im Rahmen einer Trapbehandlung abgebrochen werden.
- ○ Rechenoperationen können zu einem Interrupt führen.
- ○ Ganzzahl-Rechenoperationen können nicht zu einem Trap führen.

Aufgabe 2: `treelib` (15 Punkte)

Sie dürfen diese Seite zur besseren Übersicht bei der Programmierung heraustrennen!

Programmieren Sie eine Bibliotheksfunktion `tree()`, welche ein gegebenes Verzeichnis `pathname` rekursiv durchläuft und dabei sich selbst und alle enthaltenen Verzeichniseinträge ausgibt. Für die Ausgabe sollen Verzeichniseinträgen der bekannte relative Pfad (`pathname`) vorangestellt werden. Des Weiteren sollen Einträge ignoriert werden, wenn deren Name mit einem Punkt (".") beginnt, außer ein solcher Pfad wurde explizit an `tree()` übergeben. Zusätzlich darf `tree()` symbolischen Verknüpfungen nicht folgen um Endlosrekursionen zu verhindern.

Schnittstelle:

```
int tree(const char * const pathname);
```

Rückgabewert:

- **`0`:** die Ausgabe war erfolgreich
- **`-1`:** bei der Operation ist ein Fehler aufgetreten

Beachten Sie, dass Verzeichniseinträge, und damit die auszugebenden Pfade, beliebig lang sein können. Pfad-Konkatenationen sollen entweder mit `strcpy(3p)` und `strcat(3p)` oder mit `sprintf(3p)` durchgeführt werden. Da es sich bei `tree()` um eine Bibliotheksfunktion handelt, dürfen für dessen Implementierung ausschließlich die geforderten Ausgaben erfolgen. Da nicht alle Fehler kritisch sind (z.B. unzureichende Berechtigungen) soll auch im Fehlerfall weiterhin versucht werden, womöglich allokierte Ressourcen aufzuräumen.

```
#include <dirent.h>
#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>

int tree(const char * const pathname) {

}
```

T:

Aufgabe 3: Synchronisation (8 Punkte)

1) Erläutern Sie das Konzept Semaphor. Beschreiben Sie außerdem, welche Operationen auf Semaphoren definiert sind und was diese tun. (5 Punkte)

2) Skizzieren Sie in Programmiersprachen-ähnlicher Form, wie mit Hilfe eines zählenden Semaphors das folgende Szenario korrekt synchronisiert werden kann: Ein Hauptthread soll so lange warten, bis 7 Arbeiter-Threads ihre Arbeit (Ausführung einer Funktion `doWork()`) erledigt haben. Beachten Sie, dass es nicht notwendig ist alle freien Zeilen des untenstehenden Codefragments auszufüllen. Ihnen stehen dabei folgende Semaphor-Funktionen zur Verfügung: (3 Punkte)

- `SEM *semCreate(int);`
- `void P(SEM *);`
- `void V(SEM *);`

Hauptthread:

```
static SEM *s;
int main(void) {
  s = semCreate(0);
  start7WorkerThreads(threadFunc);
```

```
}
```

Arbeiterthread:

```
void threadFunc(void) {
```

```
  doWork();
```

```
}
```