Übungsblatt 4 – Prozesse

Aufgabe 1: Schauen Sie sich noch einmal alle Folien aus der Vorlesung an und stellen Sie sicher, dass Sie sie verstanden haben. Falls in der Vorlesung nicht alle Folien besprochen wurden, dann arbeiten Sie die nicht besprochenen Folien selbstständig durch. Benutzen Sie dabei auch die Literatur. Falls Fragen offen bleiben sollten, besprechen Sie diese während der Übungsstunde mit dem Betreuer.

Aufgabe 2:

a) Lesen Sie das Kapitel über Prozesse in dem Buch, das Sie sich besorgt haben. Die Abschnitte, in denen es in deutlich tiefere Details geht als in der Vorlesung angesprochen, können Sie überspringen.

b) Erstellen Sie Ihre persönliche handschriftliche Zusammenfassung der wichtigsten Inhalte der Lehrveranstaltung.

c) Überlegen Sie sich drei Verständnisfragen zur Lehrveranstaltung und halten diese schriftlich fest.

Wichtige Hinweise:

- Schreiben Sie alle Übungsprogramme **vernünftig formatiert** und **sinnvoll mit Kommentaren dokumentiert**.
- Bevor Sie einen Systemaufruf programmieren, **lesen Sie** in der **Dokumentation** der C-Library die Beschreibung dieses Systemaufrufs nach.
- Schreiben Sie alle folgenden C-Programme so, dass bei jedem Systemaufruf **überprüft** wird, ob ein **Fehler** aufgetreten ist. Falls Fehler auftreten, dann geben Sie eine aussagekräftige Fehlermeldung aus.
- **Schalten Sie** beim übersetzen immer alle **Compilerwarnungen an!** Benutzen Sie außerdem den Debug-Modus, so dass Sie ggf. mit einem Debugger nachvollziehen können, was ihr Programm macht.
- Automatisieren Sie alle Übersetzungsvorgänge mit Hilfe eines `Makefile` Fügen Sie ein target `clean` ein, das beim Aufruf alle Zwischen-, Hilfs- und erzeugte Programmdateien wieder löscht.
- Wenn Sie beim programmieren Fehler gemacht haben, überlegen Sie, wie Sie diesen Fehler zukünftig vermeiden, bzw. schneller entdecken können.

Aufgabe 3: Neuen Prozess erzeugen und Prozess-IDs ausgeben: Finden Sie die für diese Aufgabe relevanten Abschnitt in der Dokumentation der C-Library und lesen diese durch.

a) Schreiben Sie ein C-Programm, das
   - mit `fork` eine Kopie von sich selbst erstellt,

- im Kindprozess den Text „Kind“ und den Rückgabewert von `fork`, die eigene Prozess-ID und die Prozess-ID des Elternprozesses ausgibt,
- im Elternprozess den Text „Eltern“ und den Rückgabewert von `fork`, die eigene Prozess-ID und die Prozess-ID des Elternprozesses ausgibt.

b) Compilieren und starten Sie das Programm.

Aufgabe 4: Eine Kette von Kindprozessen erstellen. Finden Sie die für diese Aufgabe relevanten Abschnitte in der Dokumentation der C-Library und lesen diese durch.

a) Schreiben Sie ein C-Programm, das
- eine Kette von 8 Kindprozessen (Kind, Enkel, Ur-Enkel, usw.) erstellt,
- jedes Kind gibt seine Nummer in der Kette, den Text „Kind“, den Rückgabewert von fork, seine PID und die PID des Elternprozesses aus,
- der Elternprozess wartet auf das Ende seines Kindes und gibt dann seine Nummer in der Kette, den Text „Eltern“, den Rückgabewert von fork, seine PID und die PID des Elternprozesses aus.

b) Compilieren Sie ihr Programm und testen es.

c) Ändern Sie ihr C-Programm so, dass
- die Kette bis zu 50 Kindern geht,
- das letzte Kind der Kette dann auf einen Tastendruck des Benutzers wartet.

d) Compilieren Sie ihr Programm und testen es. Schauen Sie sich mit dem Kommando `pstree` den Prozessbaum an.

e) Ändern Sie ihr Programm so, dass die Kindprozesse keine Kette mehr bilden, sondern alle Kinder von demselben Elternprozess stammen also Geschwister sind. Jedes Kind soll wieder die in a) bereits genannten Informationen ausgeben.

Aufgabe 5: Gemeinsames schreiben in eine Datei.

a) Schreiben Sie ein C-Programm, das
- die Datei `abc.txt` zum lesen und schreiben öffnet. Falls sie schon existiert, soll ihr Inhalt gelöscht werden,
- mit `fork` einen Kindprozess erzeugt.
- im Kindprozess 100 000 mal des Text „Ich bin Kind“ in die Datei abc.txt schreibt.
- im Elternprozess 100 000 mal des Text „Ich bin Eltern“ in die Datei abc.txt schreibt.

b) Compilieren und starten Sie das Programm.

c) Schauen Sie sich den Inhalt der Datei an und erklären Sie sich, warum der Inhalt so aussieht.

Aufgabe 6: Prozess abspalten und darin andere Kommandos ausführen.

a) Schreiben Sie ein C-Programm, das
- einen oder mehrere Kommandonamen (ohne eigene Parameter) als Parameter erwartet.
- für jeden übergebenen Kommandonamen einen Kindprozess startet (fork), in dem dann das Kommando (mit exec) ausgeführt wird.

b) Compilieren und starten Sie das Programm.

c) Ändern Sie ihr Programm nun so, dass statt `fork, exec` und `wait` nun `system` benutzt wird.