3. Übung

Ausgabe	Abgabe	Diskussion
01.11.13	08.11.13	11.11.-15.11.13

Bitte bei der Abgabe Name/Matr.Nr. der Mitglieder einer Gruppe, Nummer der Übung/Teilaufgabe und Datum auf den Lösungsblättern nicht vergessen! Darauf achten, dass die Lösungen beim richtigen Tutor abgegeben werden. Achten Sie bei Programmieraufgaben außerdem darauf, dass diese im Linuxpool kompilierbar sind.
Zu spät abgegebene Lösungen werden nicht berücksichtigt!

Aufgabe 1: Prozesse (4*0.5=2 Punkte)

Erklären Sie den Sinn und die Funktion der einzelnen Felder im Process Control Block.

Aufgabe 2: Prozesse (2 Punkte)

Erklären Sie den `fork()`-Befehl und seine Funktionsweise.

Aufgabe 3: Prozesse (3 Punkte)

Grenzen Sie Task, Prozess und Thread voneinander ab und beschreiben Sie auch die Ressourcenverwaltung in diesen Konzepten.

Aufgabe 4: Einführung in C: Funktionen (1+2+2=5 Punkte)

Ziel dieser Aufgabe ist es, Sie mit den verschiedenen Arten von Variablenvereinbarungen, dem Gültigkeitsbereich von Variablen und der Parameterübergabe bei Funktionsaufrufen vertraut zu machen.

Grundsätzlich unterscheidet man zwischen globalen und lokalen Variablen. Globale Variablen sind im gesamten Programm gültig. Lokale Variablen können hingegen nur in der Funktion benutzt werden, in der sie deklariert wurden. Globale und lokale Variablen können gleiche Namen haben. Bei Namenskonflikten hat die lokale Variable Vorrang.

Die Parameterübergabe an Funktionen erfolgt nach dem *call-by-value*-Prinzip. Im Funktionskopf werden Variablen deklariert. Diese erhalten bei Ausführung jeweils den Wert zugewiesen, der beim Funktionsaufruf angegeben wurde. Die Variablen können dann zwar im Körper der Funktion modifiziert werden, jedoch wird der neue Wert nicht an die aufrufende Funktion zurückgegeben. Die Parameterrückgabe muss entweder explizit über den Ergebniswert der Funktion (mittels `return`) oder implizit über Speicheradressen erfolgen. Im zweiten Fall wird nicht der Wert einer Variable, sondern die Adresse ihrer Speicherstelle an die Funktion übergeben. Alle Änderungen, die an der adressierten Speicherstelle vorgenommen werden, sind dann automatisch auch in der aufrufenden Funktion sichtbar.

a) Erzeugen Sie das Programm `u3_4a` dessen Quellcode Sie auf der Homepage zur Vorlesung finden. Starten Sie es und erklären Sie die Ausgaben. Wieso haben die Variablen `a` und `b` zu unterschiedlichen Zeitpunkten verschiedene Werte? Welchen Rückgabewert hat die Funktion `diff()`?

b) Compilieren Sie nun den Quelltext zum Programm `u3_4b` und starten Sie es. Das Programm hat die Aufgabe, ein Array von Integerzahlen nach dem einfachen *bubble-sort*-Verfahren absteigend zu sortieren. Gleichzeitig soll die größte Zahl in der Variable `max` gespeichert werden. Finden und korrigieren Sie die Programmierfehler. Lassen Sie dabei die Funktionssignatur von `exchange()` unverändert.

c) Globale Variablen sollten so sparsam wie möglich verwendet werden, da sie unerwünschte Nebeneffekte hervorrufen können, evtl. Speicherplatz verschwenden und häufig eine Fehlerquelle darstellen. Verändern Sie den Quelltext von `u3_4b` und beseitigen Sie alle globalen Variablen. Sie können dazu auch die Funktionssignatur von `exchange()` modifizieren.