Aufgabe 6:

12.06.2006

job_sh (12 Punkte) Bearbeitung in Zweier-Gruppen

Programmieren Sie basierend auf der mini_sh von Aufgabe 3 eine um Hintergrundprozesse, Signalbehandlung und Job-Verwaltung erweiterte Shell: **job_sh** (**job sh**ell). Verwenden Sie dafür die Lösung (`/proj/i4sos/pub/aufgabe6/mini_sh.c)`, welche in den nächsten Tagen verfügbar ist. Die unten gestellten Fragen sind in der Dokumentation (**job_sh.txt**) zu erläutern.

a) Promptsymbol (RCS Release # 1)

Ändern Sie das Promptsymbol der Shell auf "job_sh>" und passen Sie das Makefile an die neue Aufgabe an. Legen Sie ein Unterverzeichnis für RCS an und checken Sie diese Version von jsh.c als Version 1 ein (**ci -u1 job_sh.c**). Am Ende jeder weiteren Teilaufgabe soll nun die in Klammern angegebene Version eingecheckt werden.

b) Hintergrundprozesse (RCS Release # 2)

Wenn eine Kommandozeile mit dem Zeichen & abgeschlossen wird, soll das Kommando ähnlich wie bei einer UNIX-Shell als *Hintergrundprozess* ausgeführt werden. Die Shell soll nicht auf das Terminieren dieses Hintergrundprozesses warten, sondern sofort wieder das Promptsymbol ausgeben und das nächste Kommando von der Standardeingabe einlesen.

c) Warten auf Kindprozesse (RCS Release # 3)

Was passiert, wenn Hintergrundprozesse und ein Vordergrundprozess laufen und ein Hintergrundprozess zuerst fertig wird (☞ Dokumentation)? Stellen Sie in Ihrem Programm sicher, dass garantiert immer der Exit-Status des Vordergrundprozesses ausgegeben wird. Zum Testen können Sie das Programm **sleep** als Hintergrundprozess bzw. Vordergrundprozess verwenden.

d) Signalhandler (RCS Release # 4)

Der shell-Prozess soll nun das Interrupt-Signal vom Terminal abfangen. In der Signalbehandlungsfunktion soll zuerst einmal nur die Meldung "Interrupt!" auf dem Standardfehlerkanal ausgegeben werden. (**sigaction(2)**) Was passiert, wenn Ihr shell-Programm ein Interrupt-Signal erhält und nur ein Vordergrundprozess läuft bzw. wenn auch Hintergrundprozesse laufen? (☞ Dokumentation) Ändern Sie das Programm nun so, dass die Hintergrundprozesse das Signal SIGINT ignorieren. Was hat sich dadurch am Verhalten bei einem SIGINT an die shell geändert? (☞ Dokumentation)

e) Versenden von Signalen (RCS Release # 5)

Der job_sh-Prozess soll nun zusätzlich den Sohnprozessen ein SIGQUIT schicken, wenn er selbst ein SIGINT erhalten hat (**kill(2)**).

f) Jobverwaltung (RCS Release # 6)

Implementieren Sie in der Shell ein Kommando **jobs**, das die Kommandozeilen und Prozess-Ids aller laufenden Hintergrundprozesse ausgibt. Stellen Sie sicher, dass Ihre Jobliste immer aktuelle Informationen enthält, d.h. wenn ein Hintergrundprozess terminiert, soll er auch aus der Jobliste ausgetragen werden, richten Sie hierfür einen Signalhandler für SIGCHLD ein. Geben Sie beim Terminieren eines Hintergrundprozesses auch dessen Exit-Status aus. Zur Verwaltung der Hintergrundprozesse (Jobs) verwenden Sie bitte die Implementierung aus `/proj/i4sos/pub/aufgabe6/joblist.[ch]`. (Makefile anpassen nicht vergessen!)

Abgabe: bis spätestens Mittwoch, 28.06.2006, 17:30 Uhr