Aufgabe 3: rush (15.0 Punkte)

Programmieren Sie basierend auf der vorgegebenen `clash` eine Shell, die in ihrer Funktionalität um die Umleitung von Ein- und Ausgabekanal und um Jobverwaltung erweitert ist: `rush` (**R**evamped **U**NIX **Sh**ell).
Der Kern dieser Aufgabe ist das Erkennen und Lösen von Nebenläufigkeitsproblemen im Zusammenhang mit Signalen!

a) Makefile

Erstellen Sie von der vorgegebenen Datei `clash.c` eine Kopie namens `rush.c` und schreiben Sie ein Makefile, das daraus unter Verwendung der vorgegebenen Module `plist.o` und `shellutils.o` ein ausführbares Programm `rush` erstellt. **Für die vorgegebenen Hilfsmodule finden Sie eine Doxygen-Dokumentation auf der Übungswebseite.**
Achten Sie darauf, alle Abhängigkeiten inklusive der Header-Dateien korrekt anzugeben! Das Makefile soll ohne eingebaute Regeln (`make -Rr`) funktionieren und außerdem Regeln für die beiden Standard-Pseudotargets `all` und `clean` enthalten. Greifen Sie dabei stets auf Zwischenprodukte (z. B. `rush.o`) zurück. Nutzen Sie die auf der Webseite genannten Compilerflags.

b) Umleiten des Standardein-/ausgabekanals

Die Shell soll das Umleiten der Standardein- und ausgabe von gestarteten Programmen erlauben (**dup2()** aus **dup(3p)**).
In die Standardeingabe wird der Inhalt der Datei *inFile* umgeleitet, falls das Token <*inFile* auftritt. Die Standardausgabe wird in eine Datei *outFile* umgeleitet, falls das Token >*outFile* auftritt. Dabei wird *outFile* angelegt, falls es noch nicht existiert, und überschrieben (alte Inhalte werden dabei komplett verworfen), wenn es bereits existiert. *outFile* soll standardmäßig mit Lese- und Schreibrechten für den Besitzer sowie mit Leserechten für die Besitzergruppe und alle anderen Benutzer erzeugt werden (`rw-r-r-`).
Das Umleitzeichen < bzw. > ist immer jeweils ohne Leerzeichen mit dem Dateinamen verbunden.

c) Signalbehandlung

Beim Drücken der Tastenkombination Ctrl-C (*Prozess beenden*) stellt der Terminal-Treiber der Shell und allen ihren Kindprozessen ein Signal vom Typ `SIGINT` zu. Sorgen Sie dafür, dass dieses Signal von der `rush` und von ihren Hintergrundprozessen ignoriert wird (**sigaction(3p)**) und nur von Vordergrundprozessen behandelt wird (mit Standardverhalten).

d) Sofortiges Aufsammeln von Zombieprozessen

Die `clash` sammelt angefallene Zombieprozesse jeweils erst vor der Ausgabe eines Prompt-Symbols auf. Ändern Sie dieses Verhalten so, dass Zombieprozesse immer unmittelbar nach ihrem Entstehen aufgesammelt werden, aber die Ausgabe des Ereignisses weiterhin vor dem Prompt stattfindet. Schreiben Sie hierfür eine Signalbehandlungsfunktion für `SIGCHLD` (**sigaction(3p)**) und benutzen Sie die Funktionen aus dem `plist`-Modul, um über den Zustand der Kindprozesse Buch zu führen. Beachten Sie, dass das Warten auf Vordergrundprozesse nun mittels **sigsuspend(3p)** erfolgen muss.

e) Kontrolliertes Beenden von Kindprozessen

Implementieren Sie ein Shell-Kommando `nuke [pid]`, das an den Kindprozess mit der angegebenen Prozess-ID `pid` ein Signal vom Typ `SIGTERM` zustellt (**kill(3p)**). Sollte es sich bei der PID nicht um einen Kindprozess der Shell handeln, so soll eine Fehlermeldung ausgegeben werden. Falls `nuke` ohne Argument aufgerufen wird, sollen alle Kindprozesse der Shell ein `SIGTERM`-Signal erhalten.
Im Anschluss soll die Shell eine Sekunde lang warten, um den Kindprozessen Zeit zum Terminieren zu geben (**sleep(3p)**). Stellen Sie sicher, dass kein Signal dafür sorgen kann, dass **sleep(3p)** vorzeitig zurückkehrt (**sigprocmask(3p)**)!

f) Nebenläufigkeitsprobleme

Durch den nebenläufigen Ablauf von Signalbehandlungen und dem Hauptprogramm kann es an mehreren Stellen zu sogenannten Race-Conditions kommen. Identifizieren Sie diese Stellen und lösen Sie alle potenziellen Nebenläufigkeitsprobleme in Ihrer Implementierung (**sigprocmask(3p)**). Achten Sie insbesondere auch auf Situationen, in denen die Shell Kindprozesse erzeugt bzw. beendet.

Hinweise zur Abgabe:

Erforderliche Dateien: `rush.c` (13 Punkte), `Makefile` (2 Punkte)
Bearbeitung: Dreiergruppen
Bearbeitungszeit: 11 Werktage (ohne Wochenenden und Feiertage)
Abgabezeit: 17:30 Uhr