SPiC-Aufgabe #8: `Lish`

(12 Punkte, keine Gruppen)

Entwerfen und programmieren Sie eine einfache Shell lish (**li**ttle **sh**ell) in einer Datei `lish.c`, die Programme (im Weiteren als Kommandos bezeichnet) ausführen kann.

- Ihr Programm soll als Promptsymbol den String
  `lish>`
  ausgeben.
- Die eingelesene Zeile soll in Kommandoname und Argumente zerlegt werden, wobei Leerzeichen und Tabulatoren als Trennzeichen dienen (`fgets(3)`, `strtok(3)`).
- Das Kommando soll dann in einem neu erzeugten Prozess (`fork(2)`) mit korrekt übergebenen Argumenten ausgeführt werden (`execvp(3)`).
- Die Shell soll auf das Terminieren des Prozesses warten (`wait(2)`) und den Exitstatus ausgeben. Bei der Statusausgabe soll unterschieden werden, ob der Prozess sich selbst beendet hat (`WIFEXITED`, `WEXITSTATUS`), oder ob der Prozess durch ein Signal (`WIFSIGNALED`, `WTERMSIG`) beendet wurde:

  1. Fall: Prozess beendet sich selbst (in diesem Beispiel mit Exitstatus 0):

     ```
     lish> ls -l
     ...
     Exitstatus [ ls -l ] = 0
     ```

  2. Fall: Prozess wird durch ein Signal beendet (in diesem Beispiel ein Interrupt-Signal (`SIGINT=2`) durch Drücken von `CTRL-C`):

     ```
     lish> sleep 10
     Signal [ sleep 10 ] = 2
     ```

- Nach der Ausgabe des Exitstatus soll die Shell wieder eine neue Eingabe entgegennehmen. Das Shell-Programm soll terminieren, wenn es beim Lesen vom Standardeingabekanal ein End-of-File (`CTRL-D`) erhält.
- Erweitern Sie die Shell so, dass das INT-Signal (erzeugt z. B. durch Drücken von CTRL-C) von Ihrer Shell (aber nicht von den erzeugten Kindprozessen!) ignoriert wird (`sigaction(2)`). Sie sollten nun laufende Kindprozesse durch Eingabe von CTRL-C abbrechen können, ohne jedoch dabei Ihre Shell zu beenden. Beachten Sie, dass die prozessweite Signal-Maske über einen Aufruf von `exec(3)` hinweg vererbt wird.

Hinweise:

- Ihr Programm muss POSIX-konform sein und mit folgenden Flags warnungs- und fehlerfrei kompilieren:
  `gcc -std=c99 -pedantic -Wall -Werror -D_XOPEN_SOURCE=500 -o lish lish.c`
- Sie können vereinfachend davon ausgehen, dass die Länge einer Kommandozeile maximal 1023 Zeichen beträgt. In anderen Fällen muss Ihre Shell nicht mehr korrekt funktionieren, es dürfen jedoch keine Pufferüberläufe auftreten.
- Das Programm `/proj/i4spic/pub/aufgabe8/spic-wait` eignet sich zum Testen der Reaktion auf Signale. Das Programm gibt nach dem Start seine Prozess-ID aus, sodass Sie ihm einfach mit dem Kommando `kill(1)` ein beliebiges Signal zustellen können.
- Sie können die Exitstatusausgabe testen, indem Sie das Kommando
  `/proj/i4spic/pub/aufgabe8/spic-exit`
  mit dem entsprechenden Status als Parameter aufrufen.

Abgabezeitpunkt

T01	12.07.2015	18:00:00
T02	12.07.2015	18:00:00
T03	13.07.2015	18:00:00
T04	13.07.2015	18:00:00
T05	14.07.2015	18:00:00
T06	14.07.2015	18:00:00
T07	14.07.2015	18:00:00
T08	15.07.2015	18:00:00