Aufgabe 6: fish (12 Punkte, 2er-Gruppen, Abgabe bis Mi. 1.07.09 10:00 Uhr)

Diese Aufgabe ist nur von Teilnehmern an der 4-SWS-Veranstaltung zu bearbeiten.

Entwerfen und programmieren Sie eine einfache Shell fish (**fi**rst **sh**ell) in einer Datei **fish.c**, die Programme (im Weiteren als Kommandos bezeichnet) ausführen kann.

a) Kommandoausführung und Statusausgabe

Ihr Programm soll als Promptsymbol den String *"fish> "* ausgeben. Die eingelesene Zeile soll in Kommandoname und Argumente zerlegt werden, wobei Leerzeichen und Tabulatoren als Trennzeichen dienen (**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 der Kommandoausführung 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):

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

2. Fall: Prozess wird durch ein Signal beendet (in diesem Beispiel ein INT-Signal (SIGINT=2)):

```
fish> 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.

Hinweise:

- Ihr Programm muss POSIX-konform sein und mit folgenden Flags warnungs- und fehlerfrei kompilieren:
  gcc -ansi -pedantic -Wall -Werror -D_POSIX_SOURCE -o mini_sh mini_sh.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.
- Wenn Sie in einem Terminalfenster z.B. durch Drücken von CTRL-C ein Signal auslösen, so wird dieses Signal allen Prozessen in der Prozessgruppe des Terminalfensters zugestellt, also inbesondere sowohl der fish als auch einem evtl. gerade laufenden Kindprozess.
- Das Kommando **sleep(1)** eignet sich zum Testen der Reaktion auf Signale. Sie können mit dem Kommando **ps -U $USER | grep sleep** die PID Ihres sleep-Prozesses herausfinden und diesem mit Kommando **kill(1)** ein beliebiges Signal zustellen.
- Sie können die Exitstatusausgabe testen, indem Sie das Kommando **/proj/i4spic/pub/aufgabe6/spic-exit** mit dem entsprechenden Status als Parameter aufrufen.