Aufgabe 3:

Entwerfen und programmieren Sie ein Programm **yash** (yet another shell), das ähnlich einer einfachen UNIX-Shell zum Starten von Programmen verwendet werden kann.

Ihre Shell soll dazu das Prompt **yash>** ausgeben und auf die Eingabe durch den Benutzer warten. Sobald dieser seine Eingabe mit **return** bestätigt, wird das angegebene Kommando mit allen Optionen durch Ihre Shell gestartet (**exec(2), fork(2)**). Die Shell wartet (**wait(2)**) dabei üblicherweise bis das Kommando beendet ist und fragt erst anschließend nach dem nächsten Kommando.

Durch die Eingabe von Ctrl-Z (Stop-Signal, SIGTSTOP) von der Tastatur soll der gerade im Vordergrund laufende Prozeß in den Hintergrund geschickt werden und die Shell mit dem Promptsymbol ein neues Kommando anfordern(**sigaction(2)**). Alternativ dazu kann ein Kommando auch sofort als Hintergrundprozeß gestartet werden, indem in der Eingabezeile als letztes Zeichen ein **&** angegeben wird. Ein beendender Hintergrundprozess (SIGCHLD), soll von der Shell sofort aufgeräumt werden (**waitpid(2)**).

Weiterhin existieren die folgenden Kommandos, die von der Shell direkt interpretiert werden, d.h. keine anderen Programme starten:

a) **jobs**: Die Shell gibt eine Liste aller noch laufenden Hintergrundprozesse sortiert nach ihren Startzeiten aus, wobei die Prozesse mit Nummern versehen werden sollten.

b) **fg** *[n]*: Der Hintergrundprozeß *n* wird in den Vordergrund geholt. Falls keine Nummer spezifiziert wurde, wird der zuletzt gestartete Hintergrundprozeß ausgewählt.

c) **kill** *[-signal] jobnumber*: Dem Hintergrundprozeß mit der angegebenen Jobnummer wird ein Signal zugestellt(**kill(2)**). Falls eine Signalnummer spezifiziert wurde, wird dieses Signal zugestellt, sonst standardmäßig SIGTERM.

d) **exit**: Die Shell terminiert.

Tip: Um Race-Conditions zu vermeiden ist es sinnvoll nur an einer Stelle im Programm die job-Liste zu bearbeiten und nur an dieser Stelle auf die Beendigung von Prozessen zu warten.