

Übungsblatt 2

20.01.2024/27.01.2024

Aufgabe 1 IO Streams und File Descriptors

Die drei Standard Streams sind `stdin`, `stdout`, `stderr`. Sieh auch `man stdout`.

Für einige Aufgaben ist es sinnvoll zwei Terminals zu verwenden. Dafür könnt ihr gerne `tmux` verwenden. Mit (Ctrl-b) und " oder % kann das Fenster in zwei gespalten und mit (Ctrl-b) und den Pfeiltastern kann zwischen ihnen navigiert werden.

- i. Welcher Stream wird mit `|` auf welchen anderen Stream umgeleitet?
- ii. Was ist der Unterschied zwischen `echo foo > file.txt` und `echo foo >> file.txt`?
- iii. Was passiert wenn nur `cat` ohne Argumente ausgeführt wird? Wie kann man den Zustand beenden ohne dem Prozess ein SIGINT (strg + c) zuschicken?
- iv. Wie kann `stderr` nach `stdout` umgeleitet werden?
- v. In `/usr/share/hwr/pipe_to/` befindet sich ein Programm, dass von `stdin` liest und das Gelesene in mehrere Streams schreibt.
 - a) Kopiere `/usr/share/hwr/pipe_to/` in dein *home directory* und kompiliere es indem du dich in dem Ordner befindest und `make` ausführst.
 - b) Starte das Programm mit `echo hello | ./pipe_to`. Welche Streams werden in der Shell angezeigt?
 - c) Wie kannst du alle Ausgaben von `stderr` in eine Datei `err.log` und alle Ausgaben von `stdout` in eine Datei `out.log` schreiben?
 - d) In welchen Stream schreibt `pipe_to` noch?
 - e) Mit `echo hi | ./pipe_to 42` kannst du in arbiträre Streams schreiben. Leite ihn wieder auf `stdout` oder in eine Datei um.
- vi. Öffne eine Datei und ordne ihr einen File Descriptor mit `exec 3<> datei` zu. Auf welche Arten kannst du in die neue Datei schreiben?
- vii. Erstelle eine *named pipe* mit `mkfifo <some name>`. Schreibe von einem Prozess in die pipe und lese von einem anderen Prozess aus der pipe. Wie unterscheidet sich die pipe von einer Datei?

Aufgabe 2 Lazy Professor

Lazy Professor muss eine Liste der Studierenden mit Noten (ganze Zahlen) erstellen. Die Noten gehen von 1 bis 6 und können gleichverteilt sein. Die Liste soll als CSV-Datei formatiert sein und eine Kopfzeile enthalten. Die Namen müssen alphabetisch sortiert sein (siehe `man sort`).

Name	Note
s_loechner20	2
...	1

Um zu überprüfen ob die Datei richtig formatiert ist, könnt ihr die Datei mit `sftp` herunter laden.

Aufgabe 3 Make (kein Bash)

Lazy Professor aus Aufgabe 2 will nicht immer eine neue CSV-Datei erstellen, sondern nur wenn sich die Liste der Studierenden verändert.

Schreibe ein Makefile, das das Skript aus Aufgabe 2 nur dann ausführt, wenn es eine Änderung in der Lister der Studierenden gab.

Siehe hier, um besonders wenig Namen zu wiederholen.

Aufgabe 4 sed

Jetzt wollen alle Studierenden ihre Noten in schriftlicher Form bekommen. Schreibe das Skript aus Aufgabe 2 so um, dass es für jeden Studierenden eine pdf-Datei erstellt. Es befindet sich ein `LATEX`Template unter `/usr/share/hwr/latex_template.tex`. Tauscht die Placeholder `%%NAME%%` und `%%NOTE%%` mithilfe von `sed` aus. Die pdf-Datei kann mit `pdflatex -jobname <filename> latex_template.tex` erstellt werden. Mit `pdftotext <pdf datei>` - kann der Text der pdf-Datei ausgegeben werden. Jede pdf-Datei sollte nach dem Namen des Studierenden benannt sein.

Vergesst nicht die `_` zu maskieren.