

Übungsblatt 1

06/13. Januar 2025

Aufgabe 1: Ordner und Dateien

Mach dich vertraut mit der Shell und den Befehlen `ls`, `cd`, `mv`, `touch`, `rm`, `mkdir` und `rmdir`. Dafür:

- i) Erstelle eine Datei und lösche sie wieder. Versichere dich mit `ls`, dass die Datei da war und auch wieder gelöscht wurde.
- ii) Erstelle einen Ordner und lösche ihn wieder.
- iii) Erstelle eine Datei und lösche sie wieder in einem anderen Verzeichnis.
- iv) Lösche einen Ordner, der mindestens eine Datei enthält mit nur einem Befehl.
- v) Bewege (`mv`) einen Ordner der mindestens eine Datei enthält in einen anderen Ordner. Wie kannst du das selbe mit `cp` und `rm` bewirken?
- vi) Was bewirkt `cd` und `cd -`?
- vii) Erstelle einen Ordner, wechsel in den neuen Ordner mit `cd` und lösche den Ordner. Kannst du noch Dateien anlegen mit `touch <Datei>`?

Mit `man <Befehl>`, `<Befehl> --help`, oder `help <Befehl>` könnt ihr mehr Informationen über den jeweiligen Befehl bekommen.

Aufgabe 2: Shebang

- i) Schreibe ein „Hello World“ Programm für `sh`, das sich direkt vom Terminal aus starten lässt ohne den Interpreter explizit aufzurufen.
- ii) Wiederhole die vorherige Teilaufgabe, doch verwende `python` als Interpreter. Verwende `print()`, für die Textausgabe.
- iii) Schreibe einen Wrapper um `sh`, der bevor ein Skript aufgerufen wird, den Namen der Datei ausgibt. Ändere die Shebang des Skripts aus der ersten Teilaufgabe so, dass dein eigener Interpreter verwendet wird.

Das erste Argument, dass einem Skript übergeben wird, kannst du mit `$1` erhalten.

Aufgabe 3: Bestimmte Variablen

- i) Überschreibe `$HOME` und rufe `cd` auf. Überprüfe das aktuelle Verzeichnis mit `pwd`.
- ii) Führe `echo $$` und `bash -c 'echo $$'` mehrmals aus. Was fällt auf und warum?
- iii) Führe einen Befehl aus der fehlschlägt und überprüfe den exit code mit `echo $?`. Überprüfe ihn erneut. Warum ist der Code jetzt 0?
- iv) Wie kannst du den exit code selber setzen?

Aufgabe 4: Variablen und Kontexte

- i) Setze eine Variablen in deiner Shell. Starte eine neue Shell mit `$SHELL` oder `bash`. Ist die Variable noch vorhanden?
- ii) Setze eine Variablen in deiner Shell. Schreibe ein Skript, das diese Variable ausgeben soll. Wird die Variable korrekt ausgegeben?
- iii) Was ist der Unterschied zwischen `VAR=foo bash`, `export VAR=foo && bash` und `bash` gefolgt von `VAR=foo`?

Aufgabe 5: Substitution von Befehlen

- i) Du bist zu faul immer `ls -l` zu schreiben. Speichere den Befehl in einer Variablen. Wie kannst du den Befehl in der Variablen ausführen?
- ii) Wie unterscheiden sich die folgenden Schreibweisen:
 - 1. `V="ls -l"; echo $($V)`
 - 2. `V=$(ls -l); echo $V`
 - 3. `V="ls -l"; echo "$($V)"`
 - 4. `V=$(ls -l); echo "$V"`

Aufgabe 6: Globbing

- i) Wie kannst du alle Dateien mit der Endung `.txt` im aktuellen Verzeichnis ausgeben?
- ii) Wie kannst du alle Dateien mit der Endung `.txt` im aktuellen Verzeichnis ausgeben, die mindestens drei Zeichen im Dateinamen haben?

Aufgabe 7: Kontrollstrukturen

- i) Schreibe ein `bash` Skript, das alle Dateinamen gefolgt vom Inhalt des aktuellen Verzeichnisses im Terminal ausgibt.
- ii) Ändere das Programm aus der ersten Teilaufgabe so, dass optional das Verzeichnis als Parameter übergeben werden kann.
- iii) Ändere das Programm aus der zweiten Teilaufgabe so, dass optional mehrere Verzeichnisse als Parameter übergeben werden können, die dann nacheinander ausgegeben werden.

Der Inhalt einer Datei kann mit `cat` ausgegeben werden.

Aufgabe 7: Bonus

Schreibe ein Skript, das eine Liste aller TeilnehmerInnen ausgibt, die nicht angemeldet sind. `who` zeigt an, wer angemeldet ist und unter `/usr/share/group_a/users.txt` liegt die Kursliste von Gruppe A. Der Befehl `grep` ist vermutlich ebenfalls hilfreich.