htw saar
Studiengang Kommunikationsinformatik
Prof. Dr.–Ing. Damian Weber
Dipl.-Inf. Marion Bohr
Sarah Theobald, M.Sc.

Systemmanagement und Sicherheit

4. Übung

Aufgabe 1 (Shell–Programmierung)

a) Schreiben Sie ein Shell–Skript, das vor jedes Argument den String „Hallo" setzt. Beispiel:

```
./hallo2 Peter Stefan Michael
Hallo Peter
Hallo Stefan
Hallo Michael
```

b) Schreiben Sie ein Shell–Skript `viewer`, der abhängig vom Art des Inhalts einer angegebenen Datei ein entsprechendes Programm zum Anzeigen der Datei aufruft. Falls die Datei eine Grafikdatei ist, soll beispielsweise `/usr/local/bin/xv` aufgerufen werden.

Die Unterscheidung der Inhaltstypen von Dateien können Sie treffen, indem Sie `file` aufrufen, wie im folgenden Beispiel:

```
$ file tomate.jpg
tomate.jpg: JPEG image data, JFIF standard 1.01 ...
```

Unterscheiden Sie mindestens Bilddateien (`xv`), PDF-Dateien (`xpdf`), Textdateien (`less`) und Open-Document Texte (`libreoffice`).

c) Schreiben Sie ein Shell–Skript `wavtomp3`, das WAV-Dateien in MP3–Dateien umwandelt. Hierfür können Sie `ffmpeg` benutzen.

d) Schreiben Sie ein Shell–Skript `jpgtopng`, das JPEG-Dateien in PNG–Dateien umwandelt. Hierfür können Sie `djpeg` und `pnmtopng` benutzen.

e) Schreiben Sie Shell–Skripte `counter1`, `counter2`, die für zwei Zahlen a, b als Argumente

i) alle ganzen Zahlen

ii) alle 2er–Potenzen 2^i zwischen a und b druckt.

Beispiel:

```
./counter1 17 20
17 18 19 20
./counter2 15 100
16 32 64
```

f) Lesen Sie die Manualpage von `ncal(1)`. Schreiben Sie ein Skript `late-easter`, das für als Parameter übergebene Jahreszahlen j_1, j_2 diejenigen Jahre zwischen j_1 und j_2 findet, in denen der Ostersonntag nach dem 20. April stattfindet.

g) Schreiben Sie ein Shellskript, das das Kommando `which` emuliert, siehe Manualpage `which(1)`.

h) Schreiben Sie ein Shell–Skript `countc`, das die Anzahl der im aktuellen Directory vorhandenen C–Dateien (Endung .c) ausgibt (Hinweis: `ls` und `wc` benutzen).

i) Erweitern Sie `countc` um die Fähigkeit, das Directory von der Standardeingabe zu lesen (Shellkommando `read`).

j) Erweitern Sie `countc`, das es die Ausgabe ,,zu viele" erzeugt, wenn es mehr als 10 C–Dateien sind.

k) Schreiben Sie ein Shell–Skript `chkfile`, das beliebig viele Dateien auf ihre Existenz überprüfen kann. Falls eine Datei keine reguläre Datei ist, so soll eine Fehlermeldung ausgegeben werden. Beispiel:

```
./chkfile a b c d
a existiert
b existiert nicht
c ist keine regulaere Datei
d existiert
```

l) Schreiben Sie ein Shell–Skript `good`, das abhängig von der aktuellen Uhrzeit entweder

- Guten Morgen (04:00-09:59) oder
- Guten Tag (10:00-17:59) oder
- Guten Abend (18:00-22:59) oder
- Gute Nacht (23:00-03:59)

ausgibt. Hinweis: Benutzen Sie `date` und `cut`.

m) Schreiben Sie ein Shellskript, das die Audiodatei der täglichen Sendung *Stand der Dinge* von SR1 herunterladen kann. Zur Vorgehensweise folgende Hinweise

i) Erzeugen Sie das aktuelle Datum in der Form YYYY-MM-DD. Hierzu können Sie `date` mit einer `+` Option verwenden (geeignete Variation von `date +%Y`, siehe Manualpage von `date`)

ii) das Skript lädt zunächst den XML-Feed von

`http://pcast.sr-online.de/feeds/sr1standderdinge/feed.xml`

iii) darin findet man mit `grep` die Zeile mit dem aktuellen Datum in der Form YYYY-MM-DD

iv) mit `sed` können sie die Zeile so verändern, dass nur noch der `http://....mp3` Link übrig bleibt

v) danach können Sie mit `fetch` die Audiodatei laden

vi) da dies eine SR–Abendsendung ist, empfiehlt sich bei der Vorführung oder während des Testens tagsüber das Datum des Vortages fest einzutragen. Besondere Anerkennung erhalten Sie, wenn das Skript automatisch beim Nichtvorhandensein des `http://....mp3` Links die Sendung vom Vortag lädt.

vii) jeder der Schritte i)–vi) kann fehlschlagen, Ihr Skript soll sich dann mit einer sinnvollen Fehlermeldung beenden