Brückenkurs Informatik an der FU Berlin Skript*

Daniel Faensen

12. November 1998

Inhaltsverzeichnis

*Version $Revision: 1.18$ vom $Date: 1998/11/12 14:09:58$

Vorwort

Der Brückenkurs Informatik bereitet angehende Informatikstudenten auf das erste Semester vor. Es handelt sich weder um einen Programmierkurs noch um eine allgemeine Einführung in die Arbeit mit Computern. Vielmehr soll den Studentinnen und Studenten[1] die spezifische Infrastruktur am Institut für Informatik der FU vorgestellt werden. Elementare Grundkenntnisse im Umgang mit den hier installierten Betriebssystemen

[1] Im folgenden der Einfachheit halber „Studenten“

werden vermittelt. Im Vordergrund steht die Hilfe zur Selbsthilfe: Wo finde ich Informationen? Was tun im Fehlerfall?

Dieses Skript kann im Internet eingesehen werden.[2] Es ist allerdings noch unvollständig und wird ständig weiterentwickelt. Vorliegend ist die *Revision* : 1.18.

Rückkopplung zu Skript und Lehrveranstaltung ist ausdrücklich erwünscht, sowohl positive als auch negative. Wir sind um Verbesserungen bemüht. Anregungen sollten nicht nur am Ende der Veranstaltung gegeben werden, sondern auch während der Woche. Außerdem wird der geneigte Leser gebeten, während des ersten Semesters seines Studiums zu beobachten, welche dann auftretenden Schwierigkeiten hätten vermieden werden können, wäre der Brückenkurs besser gewesen.

Der Kurs ist wie folgt aufgebaut: Nach einer allgemeinen Einführung in die zur Verfügung stehenden Systeme und einigen Worten zu den Umständen, die die Arbeit in einer Mehrbenutzerumgebung mit sich bringt, werden die Workstations des Instituts mit ihrer Fensteroberfläche vorgestellt. Den Hauptteil der Veranstaltung macht eine Einführung in den Umgang mit dem Unix-Betriebssystem aus. Der Schwerpunkt liegt hier aber klar auf elementaren Kommandos zur Arbeit mit Dateien, zum Drucken und zur Informationsgewinnung (Selbsthilfe). Informationsgewinnung ist auch Thema des nächsten Abschnitts: WWW-Seiten des Instituts zu Lehre, Hard- und Software, den Arbeitsgruppen usw. Wie komme ich an die Übungsaufgaben zu den Lehrveranstaltungen? Ein weiteres Thema: Das Senden und Lesen von EMail. Ein immer wieder thematisierter Punkt wird die Etikette sein. Der Rechnerbetrieb ist nur möglich, wenn ein vernünftiger Umgang der Benutzer mit den Maschinen und untereinander gepflegt wird. Einige Richtlinien, was unter „vernünftig“ zu verstehen ist, sollen hier gegeben werden.

1 Einführung

- Den Studenten stehen folgende Rechner zur Verfügung:
  - rund 40 Sun-Workstations, ELCs und X-Terminals unter Solaris 2.5 bzw. 2.6 (Unix) im Keller in den Räumen K 21, 23, 25 und K 63,
  - PCs unter Windows NT im Keller in den Räumen K 44, 46, 48
  - X-Terminals
- Ein Laserdrucker für ein- oder zweiseitigen Druck auf weißes oder Recyclingpapier steht im Keller im Raum K46
- Im Tutorenraum im Keller (Raum K50) steht ein Scanner zur Verfügung. Ein weiterer Scanner und ein Diabelichter stehen im Multimedia-Labor (Raum 146). Zugang für Studenten gibt es nach Absprache mit einem Dozenten.
- Welche Rechte haben die Studenten? Viele, aber:

[2] `http://www.inf.fu-berlin.de/lehre/WS98/brueckenkurs/skript`

  - keine kommerzielle und
  - keine gesetzesbrecherische Nutzung der Institutsgeräte.

  Wer eigene Software installieren möchte, kann dies in Absprache mit dem *Arbeitskreis Software* tun. Selbstverständlich sind die lizenzrechtlichen Dinge zu beachten.

- Wo sind die Hauptunterschiede zwischen DOS/Windows und Unix?
  - Unix ist ein Multi-User/Multi-Tasking-Betriebssystem, das heißt an einem Rechner können mehrere Benutzer gleichzeitig arbeiten (natürlich nicht an einem Bildschirm/einer Tastatur; man unterscheide zwischen einem *Rechner*, an den nicht zwingend ein Bildschirm angeschlossen sein muß, und Bildschirmarbeitsplätzen (X-Terminals), die u. U. nur über das Netzwerk mit einem Rechner verbunden sind) und mehrere Programme laufen gleichzeitig. Ersteres bedeutet unter anderem, daß die Unix-Rechner **nie ausgeschaltet** werden dürfen.
  - Unter Unix hat jeder Benutzer (wie bei Windows im Netz) eine eigene Kennung *(account)*, der Zugang unter seinem Namen wird durch ein nur ihm bekanntes Paßwort geschützt. Account-Anträge werden im Mathematik-Teil des Brückenkurses ausgeteilt oder sind im Keller des Infomatikgebäudes verfügbar. Abgabe beim Rechnerbetrieb, auch bekannt als *die Technik* (erster Stock). Leserlichkeit beim Ausfüllen zahlt sich aus, denn es vermeidet Mißverständnisse. Bitte beachten: Auch das initiale Paßwort soll unter Berücksichtigung der Sicherheitskriterien gewählt werden (s. Abschnitt **??**). Jeder Benutzer hat ein Heimatverzeichnis *(home directory)*, in dem er eigene Dateien ablegen kann. Von den Daten in den Heimatbereichen werden von der Technik regelmäßig Sicherheitskopien angelegt.
  - Die Systemkonfiguration unter Unix ist anders. Hat man unter DOS eine `AUTOEXEC.BAT`-Datei bzw. unter Windows ein Autostart-Verzeichnis, die die Konfiguration des Systems festlegen, so hat unter Unix jeder Benutzer (zusätzlich zu den systemweit gültigen Konfigurationen) die Möglichkeit, seine persönliche Umgebung einzurichten. Diese gilt nur für den eigenen Heimatbereich und ist auf allen Rechnern des Netzes gleich. Sie ist an jedem Unix-Rechner, der sich im Instituts-Netz befindet, gültig. Es macht also so gut wie keinen Unterschied, an welchem Rechner man arbeitet.
  - Bei Unix werden Laufwerke nicht wie bei DOS/Windows über Laufwerksbuchstaben wie `A:` oder `C:` angesprochen, sondern jedes Gerät wird wie ein Verzeichnis behandelt.

2 Sun-Workstations und CDE

Der elementare Umgang mit den Sun-Workstations unter der CDE-Oberfläche wird während der Veranstaltung vorgeführt. Zu den angebotenen Werkzeugen gehören ein

Dateimanager (FileManager `dtfile`), ein einfacher Editor zum Bearbeiten von Texten (TextEdit, `dtpad`) und ein Programm zum Lesen und Schreiben von EMail (MailTool, `dtmail`).

Wichtig sind zu Beginn die Vorgänge des *Einloggens* und *Ausloggens login, logout*. Setzt man sich an einen freien Rechner, meldet sich dieser mit der Aufforderung, seinen login-Namen einzugeben, melden. Anschließend wird man nach seinem Paßwort gefragt. Ist beides korrekt eingegeben, so präsentiert sich die CDE-Arbeitsumgebung. Eine Sitzung wird durch das Ausloggen beendet. Dies geschieht entweder durch Anwählen des Exit-Buttons unten in der Mitte des Bildschirms mit der Maus oder durch Drücken der rechten Maustaste und Anwählen des Punktes *Logout* im sich öffnenden Menü.

Achtung: Ein schwarzer Bildschirm bedeutet nicht unbedingt, daß der Rechner ausgeschaltet ist. Meist ist nur ein Bildschirmschoner aktiv. Nach Bewegen der Maus oder drücken einer Taste auf der Tastatur schaltet sich die Anzeige wieder ein. Tut sich nichts, kann der Bildschirm abgeschaltet sein. Anschalten und wieder mit Maus oder Tastatur den Bildschirmschoner deaktivieren. **Niemals** den Rechner ausschalten! (Ausnahme: X-Terminals)

Noch eine Anmerkung zur Tastatur: Diese ist – wie man schnell feststellt – bei vielen der Rechner Englisch. Daß dabei y und z vertauscht sind, ist sicher das kleinste Problem, schwerwiegender sind die fehlenden deutschen Sonderzeichen äöüÄÖÜ und ß. Wer partout nicht darauf verzichten kann, gibt sie mit Hilfe der Taste, die mit *compose* beschriftet ist, ein. Durch Drücken von (nacheinander) *compose*-`"`-a erhält man beispielsweise das ä, mit *compose*-s-s das ß. Der Rest sollte intuitiv klar sein, insbesondere die anderen Umlaute, aber auch sonstige Akzente wie é, ç oder ô erhält man auf äquivalente Weise. Das funktioniert aber nicht in allen Fenstern, zum Beispiel in der Shell (s. u.) werden diese Sonderzeichen nicht angenommen.

3 Unix-Bedienung

Der Brückenkurs ist kein Unix-Kurs. Wir werden uns hier nur mit einigen Grundlagen und den wichtigsten Kommandos beschäftigen. Als umfassende Einführung für Autodidakten findet man im WWW unter die *UNIXhelp for Users*[3] (engl.), die *Kurzeinführung in Unix*[4], oder die *Unix-Kurzanleitung*[5]. Außerdem bietet die Zentraleinrichtung Datenverarbeitung der FU (ZEDAT) entsprechende Kurse an.

Um mit den Unix-Befehlen zu arbeiten, mache man sich eine sog. *shell* auf. Dies geschieht z. B., indem man auf das Bildchen mit dem Computer am unteren Bildschirmrand mit der Maus klickt. Grob gesagt ist eine Shell ein Fenster, in dem man auf einer *Kommandozeile* Befehle eingeben kann. Am Anfang der Zeile steht der sog. *prompt*, die Eingabeaufforderung. Hier ist er standardmäßig so eingestellt, daß Benutzername und Rechnername angezeigt werden:

```
myname@computer >
```

[3] http://www.inf.fu-berlin.de/tec/software/packages/Unixhelp/
[4] http://www.chemie.fu-berlin.de/chemnet/general/unix-inf.html
[5] http://linux0.urz.uni-heidelberg.de/~mseuffer/unix.html

Hinter der spitzen Klammer > werden Eingaben, z. B. ein Programmaufruf, erwartet.

```
myname@computer > ls *.txt
```

Durch Drücken der Taste Return (oder Enter oder Eingabe) wird das Programm gestartet (der Befehl ausgeführt). Die Ausgabe des Programms erscheint ebenfalls im Shell-Fenster oder es wird (bei fensterbasierten Programmen) ein neues Fenster geöffnet.

In den folgenden Beispielen wird die Eingabeaufforderung in der Regel nicht mit angegeben.

3.1 Dateien und Dateiformate

Was eine Datei ist, davon hat wohl jeder eine zumindest vage Vorstellung. Unter Unix können Dateien *Daten* (Bilder, Texte, Daten zur Konfiguration ...) enthalten, sie können aber auch ausführbare Programme sein. Sie haben einen Namen, der unter Unix (fast) beliebig lang sein kann und auch Sonderzeichen enthalten darf. Groß- und Kleinschreibung werden (anders als bei DOS) unterschieden. Häufig aber nicht zwingend endet der Name mit einem Punkt und einigen wenigen weiteren Buchstaben, der Namens*erweiterung*. Diese ist ein Indikator für den Inhalt oder das Format der Datei. Beispiele:

```
brief.txt, letter_to_mom.text, Porträt.gif,
EinEllenlangerDateinameMitVielenSonderzeichenWie(){}_!%
```

Dateien sind in sog. Verzeichnissen *(directories)* abgelegt (s. Abschnitt 3.3). Um sich eine Liste aller Dateien eines Verzeichnisses anzeigen zu lassen, gibt man den Befehl `ls` *(list)* ein:

```
ls
```

Möchte man mehr Informationen über die Dateien erhalten, so kann man dies mit

```
ls -l
```

erreichen. Es werden dann Zugriffsrechte (s. Abschnitt 3.4), Eigentümer, Gruppe des Eigentümers, Dateigröße und Datum und Uhrzeit der letzten Änderung aufgelistet.

Zum Löschen von Dateien dient der Befehl `rm` *(remove)*:

```
rm brief.txt
```

löscht die Datei `brief.txt` ohne Rückfrage und unwiederruflich.[6] Zur Sicherheit kann man auch

```
rm -i brief.txt
```

[6] Für DOS-Dateisysteme gibt es Werkzeuge, die ein Wiederherstellen der Datei möglich machen. Unter Unix gibt es das nicht. Sollte doch einmal etwas versehentlich verlorengehen, können u. U. die Mitarbeiter der Technik Tränen trocknen, da von den Heimatbereichen regelmäßig Sicherungskopien angelegt werden.

eingeben. Der sog. Schalter `-i` weist `rm` an, *interaktiv*, d. h. mit Rückfrage zu löschen.[7] Wird mit dem FileManager gelöscht, landen die Dateien zunächst im „Papierkorb“ *(Waste)*. Von hier können sie wieder zurückgeholt werden.

(Daten-)Dateien können verschieden Typen (Graphik, Text, Video) und unterschiedliche Formate (gif-Bild, Word-Dokument, QuickTime Video) haben. Ein Indikator für das Typ und/oder Format ist die Namenserweiterung. Verschiedene Programme dienen dazu, verschiedene Formate anzusehen oder zu bearbeiten. Bilder lassen sich z. B. mit `xv` betrachten und (mit Einschränkungen) bearbeiten. Alle Klartextformate können mit `more` (oder etwas komfortabler `less`) angezeigt werden. Direkt drucken lassen sich PostScript-Dateien. Die folgende Tabelle verzeichnet für eine Auswahl von Formaten geeignete Programme.

Typ	**Format**	**Erweiterung**	**Betrachter**	**Kommentar**
Klartext (ASCII)		diverse	cat, more, less, dtpad	
	HTML	.html	netscape	WWW-Dokument
Bilder	gif-Bitmap	.gif	xv	xv konvertiert auch
	jpeg-Bitmap	.jpeg	xv	
	PostScript	.eps	xv, gv	
Druckdatei	PostScript	.ps, .eps	gv	
			lpr	zum Drucken
Komprimiert	gzip	.gz	gunzip	
	compress	.Z	uncompress	
Archiv	tar	.tar	tar -xvf	

Ist für eine Datei unbekannt, welches Format sie hat, hilft manchmal das Programm `file` weiter. Beispiel:

```
file brief.ps
PostScript document
```

Eine Kopie einer Datei wird mit dem Befehl `cp` *copy)* angelegt:

```
cp file1 file2
```

Dateien können auch unter alternativen Namen angesprochen werden. Dies erreicht man durch sog. *links*, die man mit dem Befehl `ln` anlegt. Möchte man die Datei `picture.gif` auch unter `bild.gif` ansprechen, so erreicht man dies mit

```
ln picture.gif bild.gif
```

(Allgemein: `ln` *quelle ziel*). Dieser sog. *hard link* setzt voraus, daß Quelle und Ziel auf der selben Festplatte liegen. Ist dies nicht der Fall, oder möchte man einen Link auf ein Verzeichnis setzen, so legt man einen *soft link* an:

```
ln -s Mail briefe
```

[7] Standardmäßig ist am Institut der Befehl `rm` auf interaktiv umgeschaltet.

3.2 Programme

Es gibt verschiedene Varianten, Programme unter Unix zu starten. Das CDE-Windows-System erlaubt, interaktiv mittels Maus und Menüauswahl Programme aufzurufen. Dies sind meistens selbst interaktive, Maus- und Fenster-orientierte Programme. Alternativ können Programme durch Eingabe ihres Namens auf der Kommandozeile gestartet werden. Beispiel:

```
ls
```

startet das Programm (oder Kommando) `ls`. Viele (die meisten) Programme erlauben oder erwarten eine Reihe von Parametern.

```
ls *.txt
```

listet zum Beispiel die Dateien mit der Endung `txt` auf. Der Stern, ein sog. *wildcard character* oder Platzhalter steht für eine beliebige Folge von Zeichen (`b*`: alle Dateien, die mit `b` beginnen; `b*.txt`: alle Dateien, die mit `b` beginnen und mit `.txt` enden; `*xyz*`: alle Dateien, die `xyz` irgendwo im Namen haben). `*.txt` ist ein Programmparameter. Gelegentlich sind mehrere Parameter erlaubt.

```
ls *.txt *.gif
```

listet Texte und Bilder im aktuellen Verzeichnis auf.

```
mv brief.bak brief.txt
```

benennt die Sicherungsdatei `brief.bak` in `brief.txt` um.

Das Verhalten von Programmen kann durch sog. *switches* beeinflußt werden. In der Regel werden diese Schalter durch ein vorgestelltes Minus eingegeben.

```
ls -l -a
```

listet alle Dateien im Verzeichnis auf, und zwar *ausführlich* (`-l`) und auch die versteckten Dateien (`-a`), also solche, deren Name mit einem Punkt `.` beginnt (s. Abschnitt 3.11). Der obige Befehl kann auch abgekürzt werden mit

```
ls -la
```

Manche Switches erfordern Parameter.

```
lpr -P lsu filename.ps
```

druckt `filename.ps` auf dem Drucker `lsu`. `lsu` ist der Parameter für den Switch `-P`, der `lpr` mitteilt, welchen Drucker es anzusteuern hat.

Hilfe zu Programmen/Befehlen, zu Funktion, Aufruf, möglichen Switches usw. erhält man mit `man`:

```
man ls
```

3.3 Verzeichnisse

Wie unter DOS und Windows sind unter Unix die Dateien in sog. Verzeichnissen oder *directories* organisiert. Die Verzeichnisse sind baumartig strukturiert. Es gibt ein Wurzelverzeichnis *(root directory)* /, unter das alle anderen Verzeichnisse untergeordnet sind. In der hiesigen Installation wird die nächste Ebene von u. a. folgenden Verzeichnissen gebildet:

Verzeichnis	Funktion
bin	enthält Systemprogramme
dev	Geräte (Tastatur, Maus, Festplatten, Disketten ...)
etc	Systemkonfigurationen
home	Benutzerverzeichnisse
opt	diverse Programmpakete
usr	Programme und Bibliotheken
var	Systembereich für Druck, EMail u.v.m.

Die Hierarchiestufen werden durch / getrennt (Anmerkung: unter DOS durch \). Das Heimatverzeichnis des Studenten schmidt wird z. B. durch `/home/lehrnix/schmidt`[8] angesprochen, das vorliegende Skript für den Brückenkurs liegt im Verzeichnis `/opt/htdocs/lehre/WS98/brueckenkurs/`.

Besondere Verzeichnisse sind die Heimatverzeichnisse der Benutzer. Mein eigenes Verzeichnis kann ich durch ~ ansprechen (nicht in jeder Shell möglich). Bsp.:

```
ls ~
```

listet mir den Inhalt meines Heimatverzeichnisses auf, unabhängig davon, in welchem Verzeichnis ich mich gerade befinde. Heimatverzeichnisse anderer Benutzer werden durch `~username` angegeben. Bsp.:

```
ls ~schmidt
```

Zum Wechseln zwischen Verzeichnissen dient das Kommando `cd`. Mit `cd /` wechselt man ins Hauptverzeichnis, mit `cd ~schmidt/public` in das Verzeichnis `public` des Benutzers schmidt. `cd` ohne Parameter wechselt ins eigene Heimatverzeichnis. Mit `cd ..` kommt man eine Hierarchieebene höher. (Das unter DOS übliche `cd..` ohne Leerfeld funktioniert hier nicht.) Mit dem einfachen Punkt . spricht man das aktuelle Verzeichnis an.

Man unterscheidet zwischen *absoluten* und *relativen* Verzeichnis- oder Pfadangaben. Wird ein Pfad (Verzeichnisname) in einer Form angegeben, bei der es keine Rolle spielt, in welchem Verzeichnis man sich gerade befindet, spricht man von *absoluten* Pfaden. Beispiele sind alle Pfade, die mit / beginnen. Sie gehen vom Hauptverzeichnis aus. *Relative* Pfade gehen vom aktuellen Verzeichnis aus. Befindet man sich z. B. im Verzeichnis `/usr/bin`, so ist `../lib` äquivalent zu `/usr/lib`, da `..` eine Ebene höher anspricht. `../../var` entspräche demnach `/var`. Ruft der Benutzer Schmidt im eigenen Heimatverzeichnis `ls briefe` auf, so entspricht dies `ls /home/lehrnix/schmidt/briefe`. Das

[8] `lehrnix` ist ein Rechner, auf dessen Festplatte die Heimatverzeichnisse von Studenten liegen, s. Abschnitt 3.5)

läßt sich beliebig erweitern. Bsp.: `./briefe/../../schmidt/./briefe` entspricht vorigem.

Verzeichnisse werden angelegt mit dem Kommando `mkdir` und gelöscht mit `rmdir` (nicht `md` bzw. `rd` wie unter DOS). Bsp.:

```
mkdir neues_verzeichnis
cd neues_verzeichnis
cd ..
rmdir neues_verzeichnis
```

In welchem Verzeichnis man sich gerade befindet, erfährt man durch Eingabe von `pwd`.

3.4 Dateieigentümer und Zugriffsrechte

Anders als auf den DOS/Windows-Rechnern läßt sich bei Unix für jede Datei, jedes Programm, jedes Verzeichnis kontrollieren, wer Lese-, Schreib- und (bei Programmen) Ausführungsrechte hat. Schreibrecht umfaßt auch das Recht zum Löschen. Die Rechte können in drei verschiedenen Hierarchiestufen vergeben werden: Eigentümer der Datei *(user)*, dessen Gruppe *(group)* und den Rest der Welt *(others)*. Benutzergruppen am Institut sind u. a. `institut` (Professoren, Wissenschaftliche Mitarbeiter), `staff` (Techniker), `other` und `other2` (Studenten).

Mit dem Kommando

```
ls -l
```

(`-l` steht für *long*) werden ausführliche Informationen zu Dateien und Verzeichnissen ausgegeben:

```
total 15266
drwx------   2 faensen  institut     512 Oct  1 13:05 briefe/
-rw-r--r--   1 faensen  institut 7792362 Oct  1 13:14 diplomarbeit.ps
-rw-r--r--   1 schmidt  other2      4543 Oct  1 13:16 picture.gif
drwxr-xr-x   2 faensen  institut     512 Oct  1 13:17 interna/
drwxrwxr-x   2 faensen  institut     512 Oct  1 13:16 lehre/
drwxrwxrwx   2 faensen  institut     512 Oct  1 13:13 public/
-rwxr-x---   1 faensen  other2    143608 Oct  1 13:45 rtf2LaTeX*
```

Die erste Zeile gibt den belegten Speicherplatz in diesem Verzeichnis an. Die folgende Liste enthält einen Eintrag pro Datei mit folgenden Informationen: Der erste Buchstabe gibt die Dateiart an. Ein `-` steht bei einer „normalen“ Datei, ein `d` bei einem Unterverzeichnis *(directory)*. Andere entnehme man gängigen Unix-Dokumentationen. Die folgenden neun Buchstaben geben die Benutzungsrechte wieder. Man lese sie als drei Blöcke à drei Zeichen (`rwx`). Dabei steht `r` für das *Lese*recht *(read)*, `w` für das *Schreib*recht *(write)* und `x` für das Ausführungsrecht (bei Unterverzeichnissen das Recht, in diese zu wechseln). Das erste Triplett gibt die Rechte für den *Eigentümer*, das zweite für die Gruppe und das dritte für jedermann wieder. Die Spalte mit den Ziffern 1 und 2 ignoriere man hier getrost. Es folgt der Benutzername des Eigentümers der Datei und der Name der Gruppe der die Datei zugeordnet ist. Dateigröße (in Byte) und Zeitpunkt der letzten Änderung runden die Sache ab. Demnach sind `briefe`, `interna`, `lehre` und `public` Unterverzeichnisse. Im Beispiel darf der Eigentümer in alle Verzeichnisse mit `cd` wechseln und in allen lesen und schreiben (z. B. Dateien anlegen). Im Verzeichnis `briefe` darf dies kein anderer, im Verzeichnis lehre außer ihm noch alle Mitglieder der Gruppe `institut`. Andere dürfen hier immerhin lesen. In `public` darf jedermann und jede Frau schreiben. Die beiden Dateien mit den Zugriffsrechten `-rw-r--r--` dürfen vom Eigentümer gelesen und geschrieben werden, von Gruppe und Rest nur gelesen. Das Programm `rtf2LaTeX` im Verzeichnis darf vom Eigentümer und der Gruppe other2 ausgeführt werden, nicht aber vom Rest.

Rechte vergeben werden mit dem Kommando `chmod`. Es kann auf zwei Arten aufgerufen werden, von denen hier nur die eine vorgestellt werden soll:

`chmod` *permissionlist filename(s)*

permissionlist hat die Form *benutzerklasse* $\pm$ *rechte*. Die Benutzerklassen sind `u` *(user)*, `g` *(group)*, `o` *(others)* oder auch `a` für alle, die Rechte wie die oben erläutertern `r`, `w` und `x`. Beispiele:

```
chmod g+rx briefe
```

erlaubt den Mitgliedern der Gruppe (hier `institut`), im Verzeichnis `briefe` zu lesen.

```
chmod a-w *
```

verhindert das Schreiben und Löschen aller Dateien im Verzeichnis für jeden.

```
chmod u-r diplomarbeit.ps
```

bewirkt, daß der Eigentümer die Datei nicht mehr lesen kann (selbst wenn er Mitglied der Gruppe `institut` ist). Das mag nicht sehr sinnvoll sein, ist aber vielleicht illustrativ.

Mit dem befehl `umask` wird voreingestellt, mit welchen Zugriffsrechten neu angelegte Dateien erzeugt werden. Zur Benutzung schlage man in den man-pages oder im Unix-Tutorial nach.

Neben den „normalen“ Benutzern gibt es einen Benutzer, der alles darf, den Systemverwalter *root*. Vor ihm läßt sich nichts verbergen, gilt kein Schreib-, kein Leserecht. Als root arbeiten bei uns die Techniker.

3.5 Dateisysteme im Rechnernetz

3.6 Die Shell

Man glaubt, im Zeitalter fenster-, maus-, und menüorientierter Benutzerschnittstellen bräuchte man keine Befehlsinterpreter mehr. Dem ist nicht so. Zumindest im Informatikstudium kommt man unter keinen Umständen an einem kommandozeilenbasierten

Werkzeug vorbei. Ein solches wird einem auf den Sun-Workstations unter CDE zur Verfügung gestellt, wenn man mit der Maus auf den kleinen Computer am unteren Bildschirmrand klickt.

Unter Unix gibt es verschiedene Programme, die als Kommandointerpreter oder *shell* dienen. Sie sind unterschiedlich komfortabel, und jeder erfahrene Benutzer wird seine eigene Präferenz haben. Den Benutzern am Institut wird standardmäßig die `csh` (lies: c-shell, natürlich mit englischer Aussprache) zur Verfügung gestellt. Dazu kompatibel und mit wesentlich mehr Funktionen ausgestattet ist die *tcsh*. Eine andere Shell[9] als die voreingestellte startet man durch Eingabe ihres Programmnamens, also z. B. `tcsh`.

Hauptaufgabe der Shell ist, einen eingetippten Befehl entgegenzunehmen. Er wird aufbereitet und das Programm gestartet. Die Ausgaben werden im Fenster der Shell dem Benutzer präsentiert. Das Editieren der Kommandozeile variiert von Shell zu Shell. Ausprobieren ist hier gefragt. Manchmal funktionieren die Pfeiltasten, Home und End sowie Backspace und Delete, manchmal nicht. Wichtige Hilfsmittel sind:

- Die History: Der vorige Befehle läßt sich mit `!!` wiederholen, ältere mit `!` + Anfangsbuchstaben des gewünschten Kommandos. Bsp.:

  ```
  ls *.txt
  rm brief.txt
  !l
  ```

  `!l` wiederholt den Befehl `ls *.txt`. Eine Liste der vorigen Befehle erhält man mit dem Kommando `history`.

- Hintergrundprozesse: Ein länger laufendes Programm (nicht interaktiv oder in einem eigenen Fenster laufend) kann im „Hintergrund" gestartet werden, d. h. die Shell steht, nachdem sie das Programm gestartet hat, wieder für Eingaben bereit, das Programm läuft aber weiter. Dies erreicht man durch Anhängen eines & an den Befehl. Bsp.:

  ```
  analyze huge_mount_of_data &
  netscape &
  xemacs &
  ```

  Ein Prozeß, der im Vordergrund läuft, kann unterbrochen werden durch Drücken von *control*+`Z`. Er läuft dann nicht weiter. Wieder in den Vordergrund holt man ihn durch Eingabe von `fg`. Soll der Prozeß im Hintergrund weiterrechnen, so gibt man `bg` ein. Den Abbruch eines Prozesses erreicht man mit *control*+`C`.

[9] Es gibt diverse Shells, z. B. sh, ksh, zsh, bash

- Manche Shells erlauben die Vervollständigung von Kommando- und Dateinamen durch die Tab-Taste, bei der `csh` durch ESC. Voraussetzung ist, daß die Vervollständigung eindeutig ist. Ist dies nicht der Fall, wird bei zweimaligem Tab bzw. *control*+`D` eine Liste aller passenden Namen gezeigt. Vervollständigung funktioniert auch für Benutzernamen, wenn diese mit vorangestellter Tilde (~) eingegeben wurden. Ausprobieren!

- Eine Shell wird verlassen/beendet durch Eingabe von `logout`, `exit` oder *control*-`D`.

3.7 Der Suchpfad PATH

Wo findet die Shell das gewünschte Programm? Wenn beim Kommando nicht ein expliziter Pfad (z. B. `/bin/chmod`) angegeben ist, so wird die sogenannte Umgebungsvariable `PATH` ausgewertet. Eine Umgebungsvariable *(environment variable)* ist eine Konfigurationsmöglichkeit für das Verhalten der Shell und anderer Programme. Sie wird i. d. R. in Konfigurationsdateien (s. Abschnitt 3.11) gesetzt.

`PATH` ist eine Liste von Verzeichnissen (durch : getrennt), in denen Programme/Kommandos abgelegt sind. In den von der Technik vorgegebenen Konfigurationsdateien ist ein vernünftiger Wert für den Anfang voreingestellt. Man kann ihn sich ansehen durch Eingabe von

```
echo $PATH
```

Die Reihenfolge der Verzeichnisse ist relevant. Genau in dieser werden die Verzeichnisse durchsucht. Das erste Kommando, das paßt, wird ausgeführt.

Möchte man seinen Pfad erweitern, tut man dies am besten in der Konfigurationsdatei `.cshrc` im Heimatverzeichnis (s. Abschnitt 3.13). Mit

```
set path = ( $path ~/bin )
```

sorgt man z. B. dafür, daß auch das Verzeichnis `bin` im Heimatverzeichnis durchsucht wird, wenn ein Kommando eingegeben wird.

3.8 Weitere Umgebungsvariablen

Eine Reihe weiterer Umgebungsvariablen konfigurieren das Verhalten verschiedener Programme und Werkzeuge. Einen Überblick verschafft man sich mit dem Kommando

```
env
```

Das Setzen von Umgebungsvariablen erfolgt in der csh mit `setenv`:

```
setenv PRINTER ldu
```

setzt z. B. die Variable `PRINTER` auf `ldu`, was Programme wie `lpr` anweist, standardmäßig den Drucker `ldu` anzusprechen.

Variable	sinnvolle Werte	Funktion
`PRINTER`	ldu	legt den Standarddrucker fest
`EDITOR`	vi, xemacs, dtpad	Standardeditor
`MANPATH`	/usr/man:/opt/local/man	Suchpfad für man-pages
`PATH`		Suchpfad für Programme
`PAGER`	less, more	Programm zum Blättern
`NNTPSERVER`	news.fu-berlin.de	Server für Newsgroups
`MOZILLA_HOME`	/opt/navigator	Heimatverzeichnis von Netscape
`LD_LIBRARY_PATH`		Suchpfad für Programmbibliotheken
`HOST`	demos	Name des aktuellen Rechners

3.9 Drucken

Die Drucker des Instituts sind in der Regel PostScript-fähig, d. h. sie können Dateien im PostScript-Format direkt ausgeben. Gedruckt wird mit dem Befehl

```
lpr filename.ps.
```

Dabei ist `filename.ps` der Name einer PostScript-Datei. Das funktioniert aber nur, wenn ein Standarddrucker gewählt wurde, siehe dazu Abschnitt 3.8. Die Bezeichnung `lpr` stammt noch aus alten Zeilendrucker-Zeiten und meint schlicht den *line printer*. Der obige Befehl druckt auf dem als Standard festgelegten Drucker. Mit der Option `-P printer` läßt sich ein anderer Drucker wählen. Beispiel

```
lpr -P ldu filename.ps
```

druckt auf dem Drucker `ldu`. Den Studenten stehen folgende Drucker zur Verfügung:

Name	Standort	Eigenschaften
ldu	Keller Raum K 84	Laser, doppelseitig, Recyclingpapier
lsi	— " —	Laser, einseitig, Recyclingpapier

Derselbe Drucker kann unter verschiedenen Namen angesprochen werden. Die Wahl des Namens beeinflußt z. B., ob ein- oder doppelseitig (si=simplex bzw. du=duplex)gedruckt wird.

Zum Drucken stehen eine ganze Reihe von Werkzeugen zur Verfügung. Erwähnenswert sind vor allem `a2ps`, `psnup` und `pstops`. `a2ps` (lies: a-to-ps) wandelt ASCII-Dateien (ASCII) in PostScript um und formatiert sie dabei noch recht schön. Aufruf:

```
a2ps asciifile > asciifile.ps
```

`asciifile.ps` kann anschließend mit `lpr asciifile.ps` gedruckt werden. Oder man leitet die Ausgabe direkt an das Druckprogramm weiter (siehe dazu Abschnitt 3.12):

```
a2ps asciifile | lpr
```

Mit `psnup` lassen sich mehrere Seiten auf einem Blatt verkleinert ausdrucken. Das spart Papier (schont die Umwelt) und Druck-Quotas (s. Abschnitt 3.10). Um beispielsweise eine mehrseitige DIN-A4-Vorlage `filename.ps` auf DIN-A5 verkleinert zu drucken, gibt man ein:

```
psnup -n 2 filename.ps | lpr
```

`-n 2` gibt dabei die Zahl der Seiten pro Blatt an.

Unter `~faensen/bin/` liegt ein kleines Skript `booklet`, mit dem man sich kleine DIN-A5-Heftchen erstellen kann. Der Aufruf erfolgt mit

```
~faensen/bin/booklet filename.ps
```

Die Seiten müssen nur noch sortiert werden, in der Mitte geheftet und gefaltet werden. Das funktioniert gut bei Dokumenten mit max. 40 Seiten (also nachher 10 Blättern). Voraussetzung ist, daß ein Duplex-Drucker als Standard eingestellt ist.

Die Drucker des Instituts „verstehen“ PostScript. Das bedeutet, daß andere Formate konvertiert werden müssen. Für die gängigen Graphikformate kann dazu das Program xview (`xv`) verwendet werden.

```
xv filename
```

Aus xview heraus kann man entweder mit *Print* direkt drucken, oder eine PostScript-Datei erstellen mit *Save*. Als *Format* wählt man dann *PostScript*.

Klartext-Dateien (ASCII-Format) lassen sich mit `a2ps` in PostScript konvertieren (s. o.).

3.10 Druck- und Speicher-Quoten

Jeder Student hat die Möglichkeit, in seinem Heimatverzeichnis maximal 20 MB an Daten abzulegen. Wird dieses Kontingent *(quota)* länger als eine Woche überschritten, wird dem Benutzer automatisch der erneute Zugang (login) verwehrt. Nur die Technik kann dann noch helfen. Die aktuellen Quoten werden mit dem Befehl

```
quota -v
```

angezeigt. Ausgegeben wird für den Benutzer für jedes Dateisystem, wieviele Blöcke er aktuell belegt hat (blocks), wieviele er ohne Probleme belegen darf (quota), wieviele er maximal belegen darf (limit) und wie lange er noch Zeit hat, Platz zu schaffen, bevor das für ihn erledigt wird (timeleft).

Es ist also wichtig, Plattenplatz zu sparen. Dazu gibt es verschiedene Alternativen:

1. Nicht mehr benötigte Daten löschen (`rm`).
2. Große Dateien komprimieren (`gzip filename` erzeugt die komprimierte Datei `filename.gz`, Platzersparnis häufig bis zu 70 %). Dekomprimiert wird mit `gunzip`.
3. Ganze Verzeichnisse komprimiert zusammenpacken (`gtar -cvfz tarfile.tgz directory` faßt den gesamten Inhalt von `directory` in der Datei `tarfile.tgz` zusammen und komprimiert diese). Mehr Infos erhält man mit `gtar --help`.

4. Programme davon abhalten, große Datenmengen zu speichern. Der WWW-Browser Netscape ist ein solcher Kandidat. Um die Geschwindigkeit zu optimieren, werden die zuletzt betrachteten Daten lokal in einem sog. *cache* abgespeichert. Dessen Maximalgröße ist auf 5 MB voreingestellt, also bereits ein Viertel des zur Verfügung stehenden Platzes. Wie sich das ändern läßt, steht in Abschnitt 4.2.1.

Auch für das Drucken gibt es Begrenzungen. Jeder Student darf pro Semester max. 150 Blatt drucken. Dabei zählt das Blatt Papier, doppelseitiger Druck (Duplex) zahlt sich also aus. Freie Druckkapazitäten ermittelt man mit

```
show_prq
```

Auch beim Drucken läßt sich sparen. Nützlich sind hierbei Werkzeuge wie `psnup` (mehrere Seiten auf ein Blatt) oder `booklet` (erstellt ein DIN A5-Faltblatt aus DIN A4-Vorlagen, zu finden unter `~faensen/bin/`). Näheres zum Drucken siehe Abschnitt 3.9.

3.11 Dot-(.)-Dateien

Unter Unix erfolgt die Konfiguration von System- und Heimatbereich sowie von Benutzerprogrammen in der Regel durch Klartextdateien. Insbesondere die Konfigurationsdateien in den Heimatbereichen sollen nicht bei jedem Aufruf von `ls` mit angezeigt werden. Sie sind daher „versteckt", was unter Unix erreicht wird, indem der Name der Datei mit einem Punkt . beginnt. Diese Dateien werden daher *dot files* genannt. Möchte man die Dot-Dateien in sein Listing einbeziehen, startet man `ls` mit dem Schalter `-a`:

```
ls -la
```

Wichtige Dot-Dateien sind in folgender Tabelle kommentiert:

Dot-Datei	**Funktion**
`.cshrc`	Konfiguration der Shell
`.emacs`	Konfiguration von Emacs/XEmacs
`.history`	Die letzten eingegebenen Kommandos
`.login`	Kommandos, die beim Login ausgeführt werden
`.logout`	Kommandos, die beim Logout ausgeführt werden
`.netscape`	Keine Datei, Verzeichnis, in dem Netscape seine Konfigurationen, seinen Cache usw. ablegt
`.signature`	Eine Art Visitenkarte, die an jede EMail angehängt wird (Adresse usw.)
`.X*`	Verschiedene Dateien zur Konfiguration der Fenster-Oberfläche

3.12 Ein-/Ausgabeumleitung, Pipes

Die Ausgabe eines Kommandos läßt sich in eine Datei umleiten. Dazu dient die spitze Klammer `>`. Bsp.:

```
ls -l > listing
```

schreibt das Ergebnis von `ls -l` in die Datei `listing`. Eine etwa bereits bestehende Datei `listing` würde dabei überschrieben. Mit einer doppelten spitzen Klammer `>>` kann man die Ausgabe an eine bestehende Datei anhängen.

```
ls -l >> listing
```

Ebenso läßt sich die Eingabe an ein Programm statt von der Tastatur aus einer Datei einlesen. Bsp.:

```
mail schmidt@inf.fu-berlin < brief_an_schmidt
```

gibt den Inhalt von `brief_an_schmidt` an `mail` weiter.

Es ist auch möglich, die Ausgabe des einen Kommandos direkt an ein anderes Kommando als dessen Eingabe weiterzuleiten.

```
mein_programm | mail tutor
```

gibt die Ausgabe von `mein_programm` an das Kommando `mail`. Dieses sendet sie per EMail an den Tutor namens `tutor`. Mehrere Pipes können hintereinandergeschaltet werden:

```
ls -la | a2ps | lpr
```

listet das Verzeichnis auf und gibt das Ergebnis an ein Programm zur Formatierung und Umwandlung von Klartext in die Druckersprache PostScript. Dieser PostScript-Code wird dann an das Druckprogramm weitergeleitet.

3.13 Konfiguration des Heimatbereichs mit .cshrc

In diesem Abschnitt wird die von der Technik vorgegebene Datei `.cshrc` ausführlich behandelt.

Die Datei `.cshrc` sieht wie folgt aus (Die Zeilennummern wurden mit dem Kommando `nl` hinzugefügt):

```
# .cshrc Prototyp
#
# set up search path
setenv OPENWINHOME /usr/openwin
set path = (/bin /usr/bin $OPENWINHOME/bin)
setenv PAGER less
setenv LC_CTYPE iso_8859_1
setenv NNTPSERVER news.fu-berlin.de
alias a         alias
#
# define Operating System
```

```
setenv OS "`uname -r | cut -f1 -d'.'`"
if ( $OS == 4 ) then
   #    BSD-System
   # ****** HOMES  *********
   setenv XHOME /usr/local/X11
   setenv NPHOME /usr/local/newsprint
   setenv FMHOME /usr/local/frame
   # ****** ENV's ************
   setenv LD_LIBRARY_PATH /usr/lib/X11:$OPENWINHOME/lib:/usr/lib
   setenv MIRALIB /usr/local/mira/miralib
   set path = ( /usr/local/gcc/bin $path /usr/ucb /usr/hosts /usr/lang \
                /usr/local/mira . $XHOME/bin \
                /usr/local/bin $NPHOME/bin $FMHOME/bin \
                /usr/local/emacs/bin)
   source ~staff/bin/skripts/setman        # MANPATH
   source ~staff/bin/skripts/absolve.csh   # AnswerBook + SunSolve
   source /usr/local/tex/texsetup.csh      # tex Umgebung
   # ******* CMD_ALIASES ***********
    a ll        'ls -lag'
    a r         reset
else
   #    SYSTEM-5
   # *******  HOMES ****************
   setenv MOZILLA_HOME /opt/navigator
   # *******  ENV's ****************
   setenv XFILESEARCHPATH /usr/openwin/lib/locale/%L/%T/%N%S:/usr/openwin/lib/%T/%N%S:/opt/local/lib/%T/%N
   setenv MIRALIB /opt/mira/miralib
   setenv LD_LIBRARY_PATH /usr/dt/lib:$OPENWINHOME/lib:/usr/lib:/usr/ucblib:/opt/local/lib:/opt/gcc/lib
   set path = ( /usr/dt/bin $path \
                /opt/SUNWspro/bin /usr/ccs/bin \
                /opt/gcc/bin /opt/local/bin /usr/ucb \
                /opt/netscape /opt/navigator /opt/tex2e/bin \
                /opt/mira . /opt/emacs/bin )
   source ~staff/bin/skripts/setman.5      # MANPATH
   source ~staff/bin/skripts/absolve5.csh  # AnswerBook + SunSolve
   # ******* CMD_ALIASES ***********
    a ll        'ls -la'
endif
#
set noclobber
limit coredumpsize 0
set filec

# aliases for all shells
```

```
55  a cd    'cd \!*;echo $cwd'
56  a cp    'cp -i'
57  a mv    'mv -i'
58  a rm    'rm -i'
59  a pwd   'echo $cwd'
60  a del   'rm -i'

61  # skip remaining setup if not an interactive shell

62  if ($?USER == 0 || $?prompt == 0) exit

63  # settings for interactive shells

64  stty erase "^H"
65  set history = 40
66  set savehist = 40
67  setenv HOST "`uname -n`"
68  set prompt = "${USER}@${HOST} \!: "

69  # other aliases

70  a h             'history -r | more'
71  a type  cat
72  a lock  xlock
73  a m     more
74  a list  less
75  a dir   ls
76  a la    'ls -a'
```

Die Zeilen 1 bis drei beginnen mit einem `#`. Das kennzeichnet sie als Kommentar, der Text hinter dem `#` wird beim Ausführen der `.cshrc` ignoriert.

In Zeile 4 wird die Umgebungsvariable `OPENWINHOME` gesetzt. Sie gibt für die Arbeitsumgebung OpenWindows (s. Abschnitt 3.16), wo es seine Programme, Bibliotheken, Schriften, Bildchen usw. zu suchen hat. In Zeile 6 wird der Suchpfad gesetzt (s. Abschnitt 3.7). Dabei wird bereits die Umgebungsvariable `OPENWINHOME` benutzt. Vollständig wäre diese Zeile als

```
set path = (/bin /usr/bin /usr/openwin/bin)
```

zu lesen. In den Zeilen 6 bis 8 werden weitere Umgebungsvariablen gesetzt. Zu deren Bedeutung siehe Abschnitt 3.8.

In Zeile 9 steht der Befehl `alias`. Er dient dazu, einem (mehr oder weniger koimplexen) Kommando einen neuen, meist einfacheren Namen zu geben. `alias a alias` im Beispiel sorgt dafür, daß mit `a` der Befehl `alias` aufgerufen wird. Anwendungen von `alias` sind in den Zeilen 48, 55–60 und 69–76 zu finden. `alias ll 'ls -l'` (Zeile 48) sorgt z. B. dafür, daß an Stelle des Befehls `ls -l` auch einfach `ll` eingegeben werden kann.

Zeile 55 verdeutlicht die Umbenennung eines komplexen Kommandos

```
alias cd    'cd \!*;echo $cwd'
```

bewirkt, daß mit `cd` *dir* nicht nur in das Verzeichnis *dir* gewechselt, sondern anschließend auch noch die Umgebungsvariable `cwd` *(current work directory)* ausgegeben wird. `\!*` wird durch den Parameter für `cd` (im Beispiel *dir*) ersetzt.

In Zeile 12 wird mit einem komplexen Kommando ermittelt, ob man sich gerade auf einem Rechner mit dem Betriebssystem SunOS 4 oder 5 befindet (SunOS 5 ist das gleiche, wie Solaris 2). Dies wird der Umgebungsvariablen `OS` zugewiesen. Von der Betriebssystemversion ist das weitere Vorgehen abhängig. In Zeile 13 wird entschieden, ob die Zeilen 14 bis 31 (bei Version 4) oder 33 bis 48 ausgeführt werden. Da am Institut nur noch ein einziger Rechner unter SunOS 4 läuft (`puma`), wird auf die Erläuterung der ersten Alternative verzichtet.

Die Umgebungsvariablen, die hier gesetzt werden, sind – soweit relevant – in Abschnitt 3.8 erläutert.

In den Zeilen 45 und 46 werden mit dem Kommando `source` Zwei Skripten (ähnlich der gerade diskutierten `.cshrc`) geladen und die darin enthaltenen Kommandos ausgeführt.

Das Setzen der Variablen `noclobber` in Zeile 51 verhindert, daß durch Umleiten der Standardausgabe mit `>` existierende Dateien überschrieben werden (s. Abschnitt 3.12).

Wenn Programme „abstürzen“ wird unter Unix zur anschließenden Fehlersuche ein Abbild des Hauptspeichers, ein sogenannter *core dump*, auf die Festplatte geschrieben. Dieser kann recht groß sein und, wenn er unbemerkt bleibt, die Festplattenquotas des Benutzers erschöpfen. Daher wird in Zeile 52 die maximale Größe eines core dumps auf 0 gesetzt.

Durch Setzen der Variable `filec` (Zeile 53) kann die `csh` Datei- und Benutzernamen vervollständigen. Siehe dazu Abschnitt 3.6

Zeile 64 stellt die Shell so ein, daß man mit der Backspace-Taste rückwärts löschen kann.

Zeile 65 und 66 stellen ein, daß die letzten 40 Befehle zum erneuten Aufruf erhalten bleiben.

In Zeile 67 wird die Umgebungsvariable `HOST` auf den Namen des aktuellen Rechners gesetzt. Dazu wird da Kommando `uname -n` aufgerufen. Da es in nach links stehenden einfachen Anführungsstrichen steht, wird es zuerst ausgewertet, sein Ergebnis (seine Ausgabe) wird der Variablen zugewiesen.

In Zeile 68 wird die Eingabeaufforderung, der *prompt*, festgelegt. Es ist eine Kombination aus den Inhalten der Variablen `USER` und `HOST`, getrennt durch den Klammeraffen, gefolgt von der Nummer des aktuellen Befehls (für den Historymechanismus) und einen Doppelpunkt. Der Prompt kann nach Geschmack verändert werden.

Bei Änderungen in der `.cshrc` bietet es sich an, den ursprünglichen Eintrag zu erhalten aber durch Voranstellen eines `#` auszukommentieren und damit von der Abarbeitung auszuschließen.

3.14 Arbeiten im Netz

Vernetzung und Mehrbenutzerbetriebssysteme laden ein, nicht nur an dem Rechner zu arbeiten, an dem man sitzt, sondern andere Maschinen mitzubeschäftigen. Insbesondere wenn der Comuter vis-a-vis ein älteres Modell ist, weicht man gern auf einen schnelleren aus. Außerdem sind bestimmte Programme nur auf speziellen, diesen Programmen gewidmeten Rechnern lauffähig. Der sogenannte *remote login* auf einen anderen Rechner erfolgt so (Annahme: `hai` ist ein langsamer Rechner, an dem man sitzt, `ultra` ein schneller, auf dem man ein Ressourcen-fressendes Programm laufenlassen möchte.):

```
myname@hai: > xhost +ultra
myname@hai: > rlogin ultra
myname@ultra: > setenv DISPLAY hai:0
myname@ultra: > emacs &
```

`xhost +ultra` erlaubt, daß Programme, die auf `ultra` laufen, `hai` für die Ausgabe nutzen. Hat man sich auf `ultra` eingeloggt, muß man diesem Rechner noch mitteilen, welcher Rechner für die Ausgabe genutzt wird. Dies geschieht durch setzen der Umgebungsvariablen `DISPLAY`.

Aber: Beachte dazu die Anmerkungen zur Etikette (Abschnitt 8).

Es sei außerdem auf die Sicherheitsprobleme beim rlogin hingewiesen (s. Abschnitt 9). Statt `rlogin` verwende man `ssh`.

3.15 Disketten

Unter Unix kann man nicht wie unter DOS/Windows einfach auf Knopfdruck eine Diskette aus dem Laufwerk holen. Bei den Sun-Workstations geschieht der Auswurf mit dem Befehl `eject`.

Um Dateien von einer Diskette auf das Unix-Dateisystem (oder umgekehrt) zu kopieren, bedient man sich des Programms `mcopy`. Mit

```
mcopy filename a:
```

kopiert man die Datei `filename` auf Diskette,

```
mcopy a:*.txt ~/texte
```

kopiert alle Dateien mit der Endung `txt` in das Verzeichnis `texte`. `mcopy` gehört zu einer Sammlung von Werkzeugen zur Bearbeitung von DOS-Dateien und Disketten, den sog. `mtools`. Informationen erhält man mit `man mtools`.

3.16 Alternative zu CDE: Open Windows

4 Informationsgewinnung

4.1 Manual Pages

Zu (fast) jedem Unix-Kommando existiert eine sog. *man page*, also eine Art elektronischer Auszug aus einem Anwenderhandbuch. Mit dem Befehl

```
man commandname
```

erhält man diese Seite, die neben einer Beschreibung des Kommandos alle möglichen *switches* erläutert. Ferner findet man manchmal Beispiele und Verweise auf verwandte Befehle. Die *man page* für das Kommando `man` erhält man selbstverständlich mit

```
man man
```

Nicht immer kennt man den genauen Befehl für ein Programm. Hilfe bietet dann `apropos`. Mit

```
apropos keyword
```

erhält man eine Liste von Kommandos, die `keyword` in ihrer Kurzbeschreibung haben. Hat man den Befehl zum Kopieren von Dateien vergessen, so liefert

```
apropos copy
```

eine (lange) Liste, die unter anderem `cp` und `mcopy` enthält.

4.2 WWW

Das Internet ist ein Netz von mehreren Millionen Rechnern weltweit. Es umfaßt eine Reihe von Diensten zur Kommunikation zwischen diesen Rechnern. Dazu gehören neben vielen anderen EMail, die Newsgroups (s. Abschnitt 4.3) und das World Wide Web, kurz WWW oder einfach *das Web*. Das WWW ist aus den Medien und vielleicht aus eigener Erfahrung wohlbekannt. Daß oft Internet und WWW (fälschlicherweise) synonym gebraucht werden, ist ein Zeichen dafür, wie stark das WWW hier dominiert.

Die Basis des WWW ist, daß die Dokumente durch Verweise *(links)* aufeinander vernetzt sind. Diese Tatsache verliert aber zunehmend an Bedeutung, und das Web dient mehr und mehr als Plattform

- zur Veröffentlichung von Dokumenten zur Selbstdarstellung (Home Pages), um Informationen zu verbreiten oder auch Programmdokumentationen verfügbar zu machen,
- als Schnittstelle zu großen Informationsquellen und zum
- elektronischen Kommerz.

Die Sprache des Webs sollte HTML sein, ein Format, in dem die Web-Dokumente angelegt sind. Zum Zugriff auf das WWW und zur Darstellung von HTML-Dokumenten benötigt man eine geeignete Software, einen *browser*. Die verbreitetsten Web-Browser sind der Netscape Navigator und der Microsoft Internet Explorer. Web-Seiten werden über eine Adresse in der Form angegeben (Bsp.):

`http://www.inf.fu-berlin.de/lehre/index.html`

Dabei ist `http` das Protokoll (ein anderes ist z. B. `ftp`). `http://` kann bei den meisten Browsern auch weggelassen werden. `www.inf.fu-berlin.de` ist der Name des Rechners (des Web-Servers), auf dem die Web-Seite liegt. `lehre` ist hier das Verzeichnis und `index.html` der Name des Dokuments. Die Adresse bezeichnet man als *URL*. Eine Web-Adresse erreicht man entweder durch anklicken von blau markierten Verweisen auf diese Adresse in anderen Dokumenten (das sprichwörtliche *Surfen*) oder durch Eingabe der Adresse in einem Eingabefeld des Browsers.

Eine eigene Homepage erstellt man übrigens durch Anlegen eines Verzeichnisses `public_html` im eigenen Heimatverzeichnis. In diesem muß eine Datei `index.html` liegen. Diese ist die Einstiegsseite, die man unter dem URL `http://www.inf.fu-berlin.de/~myname` erreicht. Sowohl `public_html` als auch `index.html` müssen für jedermann lesbar sein (s. Abschnitt 3.4).

```
chmod o+rx ~/public_html
chmod o+r ~/public_html/index.html
```

4.2.1 Netscape-Konfiguration

Zum Zugriff auf das WWW benötigt man ein Programm, das einem die geladenen Seiten darstellt, einen sog. `browser`. Netscape ist einer der beiden verbreitetsten Browsern. Hier installiert sind die Versionen Navigator 3.0, Navigator 4 und Communicator. Sie befinden sich in den Verzeichnissen

Version	Verzeichnis	Speicherbedarf	unterstützt
Navigator 3.04	/opt/netscape	4,8 MB	WWW, EMail, News
Navigator 4.0	/opt/navigator	6.5 MB	WWW
Communicator	/opt/communicator	11,5 MB	WWW, EMail, News

Der Speicherbedarf der obigen Tabelle bezieht sich nur auf das ausführbare Programm. In der Realität ist er wesentlich höher. Abhängig davon, welches der in der Tabelle angegebenen Verzeichnisse im Suchpfad (s. Abschnitt 3.7) eingetragen ist, wird die entsprechende Browser-Version beim Aufruf von Netscape gestartet. Wie unschwer zu erkennen, nimmt der Ressourcenbedarf mit der Browser-Version stark zu. Es ist daher insbesondere auf den eher schwächeren Maschinen die schlanke Version 3.0 vorzuziehen, auch wenn diese nicht alle Neuerungen im WWW unterstützt. Die Version 4.0 ist noch lauffähig, bietet aber leider keinen EMail- und News-Client. Der Communicator ist das angemessene Werkzeug, wenn man seinen Rechner lahmlegen möchte. Noch effizienter läßt sich dies mit dem Internet Explorer erledigen, der ebenfalls für die Suns installiert ist.

Konfiguriert wird der Navigator 3.04 über das Menü *Options*. Die folgenden Anweisungen seien beispielhaft verstanden.

1. *General Preferences*: Unter *Appearance* trage man als *Home Page Location* den URL des Instituts ein: `http://www.inf.fu-berlin.de`. *Fonts, Helpers und Images* lasse man unverändert. Die *Helpers* sorgen dafür, daß bei bestimmten Dokumenten, die man aus dem Internet lädt, gleich automatisch das entsprechende Anwendungsprogramm gestartet wird, das die Daten anzeigen kann. Man wähle *application/postscript*, klicke auf den Schalter *Application* und trage ein `gv %s`. Das sorgt dafür, daß ein aus dem Internet geladenes Postscript-Dokument gleich mit Ghostview dargestellt wird. Auprobieren kann man dies direkt an diesem Skript, indem man Netscape anweist, den URL `http://www.inf.fu-berlin.de/lehre/WS98/brueckenkurs/skript.ps` zu laden. Mit *Ok* bestätigen. Die gleiche Prozedur für *application/pdf* wiederholen. Hier wird als *Application* `acroread %s` eingetragen. `acroread` startet den Acrobat Reader von Adobe für PDF-Dokumente. Beide *Helper* sind u. U. nötig, um auf Übungsaufgaben, Skripten u. ä. zuzugreifen.

2. *Editor Preferences*: Nur nötig, wenn mit Netscape eigene HTML-Seiten erstellt werden sollen.

3. *Mail and News Preferences*: Möchte man Netscape als EMail und/oder News-Client einsetzen, sind hier einige Konfigurationen vorzunehmen. Unter *Servers* stellt man als *SMTP Server* `fubinf.inf.fu-berlin.de` ein, das gleiche als *POP3 Server* (*Built in Movemail* könnte auch funktionieren). Der *NNTP Server* ist `news.fu-berlin.de`. Unter *Identity* stellt man seinen Namen und seine EMail-Adresse ein. Das *Signature File* ist eine Datei, deren Inhalt an jede EMail angehängt wird.

4. Die *Network Preferences* sind hervorzuheben, denn hier stellt man den Cache ein. Ganz wichtig ist es, den *Disk Cache* auf 0 KB zu setzen. Standardmäßig sind 5000 KB voreingestellt, womit man einen Großteil seines verfügbaren Plattenplatzes (s. Abschnitt 3.10) bereits belegt. Der Cache verhindert das erneute Laden von Seiten aus dem Netz, die bereits einmal geladen worden sind. Unter den *Protocols* kann man einstellen, ob man gewarnt werden möchte, wenn ein Cookie gesetzt werden soll (s. Abschnitt 9.1). Java und JavaScript, ebenfalls Sicherheitslücken im Internet, können unter *Languages* abgestellt werden.

Beim Navigator erreicht man die Konfigurationseinstellungen über das Menü *Edit – Preferences*. Das für die Version 3.0 gesagte sollte hierauf übertragbar sein.

4.2.2 WWW-Seiten des Instituts

Der Web-Server des Institut ist wie gehabt unter `www.inf.fu-berlin.de` erreichbar. Dort finden sich u. a. Informationen zur Organisation, zu den Arbeitsgruppen, Forschungsprojekten[10], Hinweise zur Technik[11] und die Web-Seiten zu den Lehrveranstaltungen[12]. Zur Zeit werden die Web-Seiten überarbeitet.

4.3 Newsgroups

Nicht nur zur Informationsgewinnung sondern auch zur themenorientierten Diskussion mit Internetteilnehmern weltweit dienen die Newsgroups. Mehrere Tausend dieser Diskussionsgruppen mit Themenspektren von Magersucht über Kosovo-Krise, Filmbesprechungen, Religion, gängige Betriebssysteme und Anwenderprogramme stehen zum Abonnement bereit. Für Studenten am Institut sind vor allem die Newsgroups

`bln.announce.fub.cs` und `bln.announce.fub.cs.d`

von Interesse. `bln` steht dabei für Berlin, `fub` für (wer ahnt's?) Freie Universität Berlin, `cs` für Computer Science. `bln.announce.fub.cs` ist eine moderierte Newsgroup, `bln.announce.fub.cs.d` hingegen unmoderiert, das `d` steht für *discussion*.

In diesen Newsgroups können alle Interessierten (weltweit!) kurze Nachrichten *posten*. Es gehört dabei zum guten Ton, erst einmal einige der Artikel zu lesen, einigen Diskussionen zu folgen, bevor man selbst seinen Senf dazu gibt. So vermeidet man Peinlichkeiten, wie z. B. das Stellen einer „Häufig Gestellten Frage“ (FAQ, *Frequently Asked Question*) oder die Wahl der falschen Newsgroup.

Zum Üben und Ausprobieren dienen die Newsgroups `bln.test` und `de.test`.

Programme, mit denen man Newsgroups lesen und in ihnen veröffentlichen kann, gibt es viele. Netscape hatte in der Version 3 noch einen News-Reader im Navigator integriert. Seit der Version 4 hat nur noch der Communicator einen News-Reader, hier unter dem Namen Collabra Discussions verborgen. Empfohlen werden sollen hier `xvnews` und `xrn`. Diese sind Fenster-basiert. Ein News-Reader, der ohne Fensteroberfläche auskommt und daher auch bei Terminalverbindungen eingesetzt werden kann ist `tin`. Leider ist der Suchpfad zu `tin` standardmäßig nicht definiert. Es muß daher entweder durch angabe des absoluten Pfades gestartet werden

```
/usr/local/public/bin/sol/tin
```

oder man erweitert seinen Suchpfad (am besten in der `.cshrc`) um `/usr/local/public/bin/sol/`:

```
set path = ( $path  /usr/local/public/bin/sol/tin )
```

Anschließend kann `tin` direkt gestartet werden.

[10] `http://www.inf.fu-berlin.de/inst/`
[11] `http://www.inf.fu-berlin.de/tec/`
[12] `http://www.inf.fu-berlin.de/lehre/`

5 EMail

Es gehört inzwischen zum Menschen von Welt, auf seiner oder ihrer Visitenkarte eine EMail-Adresse angeben zu können. Dem Studenten für Informatik an der FU Berlin erhält eine EMail-Adresse *(mail account)*. Der Aufbau einer EMail-Adresse ist vertraut: ein Benutzername gefolgt vom Symbol @, das von englischen Kaufleuten benutzt wurde, um Stückpreise anzugeben (4 items @ 10 p), und angeblich daher als „at" gelesen wird, vermutlich aber einfach deshalb gewählt wurde, weil es mit an Sicherheit grenzender Wahrscheinlichkeit nicht Bestandteil irgendeines Namens ist. Im Deutschen wird @ oft als Klammeraffe bezeichnet. Hinter dem Klammeraffen folgt die Institution oder der Rechner, zu dem der Adressat gehört. Der Student Schmidt ist demnach erreichbar unter

`schmidt@inf.fu-berlin.de`

Innerhalb des Instituts kann der Rattenschwanz hinter dem login-Namen auch weggelassen werden.

Zum Senden und Lesen von EMail gibt es viele verschiedene Programme *(mail clients)*. Von diesen sollen `mail`, `elm` und `Mailer` (unter CDE) kurz vorgestellt werden. Andere Clients sind Pine, Netscape (Navigator Version 3 oder Communicator, nicht aber Navigator 4) oder der Emacs. Welcher Client zu bevorzugen ist, ist größtenteils Sache des Geschmacks und der Gewohnheit. Gern benutzt wird auch `mutt`.

Zum Ausprobieren, ob das Senden und Empfangen funktioniert, kann eine EMail an `echo@tu-berlin.de` gesendet werden. Man erhält „postwendend" eine Kopie des gesendeten zurück.

Noch ein paar Anmerkungen zur Terminologie:

Reply Alle EMail-Clients unterstützen das direkte Antworten auf eine eingegange EMail. Als Adressat wird automatisch der Absender gewählt. Oft wird dabei die EMail, auf die geantwortet wird, kopiert und z. B. mit > markiert. Man kann dann direkt auf Aussagen des Adressaten Bezug nehmen.

Forward oder Zoom Eine eingegangene EMail wird an einen weiteren Adressaten weitergeleitet.

Attachment Mit einer EMail kann man Dateien als Anhang mitsenden. Dies können einfache Texte sein, aber auch Binärdateien wie Programme, Graphiken oder Texte in einem bestimmten Format wie z. B. WinWord. Es ist jedoch nicht immer sichergestellt, daß der Empfänger den Anhang ordnungsgemäß empfangen kann, da nicht alle EMail-Clients die selbe Methode des Verpackens verwenden.

Mail-Verteiler Ein Mail-Verteiler ist eine EMail-Adresse, unter der man eine ganze Reihe von Adressaten erreichen kann. Manche Verteiler sind manuell gepflegt, bei anderen kann man sich als eine Art Abonnent eintragen. Am Informatikinstitut gibt es die Verteiler `stud` (alle Studenten), `inst` (Professoren und Mitarbeiter), `tut` (Tutoren) und `staff` (Mitarbeiter der Technik). Bei ernsten Rechnerproblemen schicke man eine EMail an `staff`. Ansonsten sind die Verteiler mit Vorsicht zu benutzen. Siehe dazu auch Abschnitt 8.

Nicht alle EMail-Clients sind in der Lage, Sonderzeichen wie Umlaute und ß ordentlich darzustellen. Es ist daher freundlich gegenüber seinem Addressaten, auf diese zu verzichten und sich stattdessen mit ae, oe, ue und ss zu begnügen, sofern dies möglich ist.

5.1 CDE Mailer

Für den Anfänger sicher am einfachsten zu bedienen ist der Mailer unter CDE. Man ruft ihn auf durch Klicken auf den Kasten mit Briefumschlägen am unteren Bildschirmrand.

5.2 elm

elm hat gegenüber dem CDE Mailer den Vorteil, daß es auch ohne Fensteroberfläche funktioniert. Es ist daher ein nützliches Werkzeug, wenn man über eine Terminalverbindung von zuhause oder einem anderen Arbeitsplatz mit dem Institut verbunden ist.

Nach dem ersten Aufruf mit

```
elm
```

wird ein (verstecktes) Verzeichnis `~/.elm` angelegt. Anschließend kann gearbeitet werden. Die möglichen Kommandos stehen im unteren Teil des Bildschirms. Doch Vorsicht! Standardeinstellung für den Editor zum Erstellen von EMails ist der vi (s. Abschnitt 7), der gerade für Anfänger nicht sehr intuitiv zu bedienen ist. Mit `o` (Options) läßt sich dies umstellen, z. B. auf xemacs, emacs oder pico (ein sehr einfach zu bedienender Editor, der eigentlich zum mail client pine gehört).

5.3 mail

`mail` ist vornehmlich kommandozeilenorientiert und soll genau deshalb hier nicht unerwähnt bleiben. Es hat den Vorteil, daß Ausgaben von Programmen direkt hierhin umgeleitet werden können (geht auch mit elm). Zur Ausgabeumleitung siehe 3.12. Manchmal geht das schneller und einfacher als mit *attachments*. So kann z. B. die Ausgabe eines Programms `mein_programm` mit

```
mein_programm | mail tutor
```

direkt an den Tutor mit dem Namen `tutor` gesendet werden. Das Versenden einer (nicht binären!) Datei kann mit

```
mail < filename
```

erfolgen.

Beim interaktiven Lesen und Schreiben ist `mail` eher schwerfällig und daher nicht zu empfehlen.

6 Der Editor Emacs

Unter Unix stehen eine ganze Reihe von Editoren zur Verfügung. CDE TextEdit ist sicher sehr einfach zu bedienen, aber nicht sehr vielseitig. Eine „eierlegende Wollmilchsau“ unter den Editoren ist der Emacs. Es ist ein sehr flexibel konfigurierbarer Editor für Programmierarbeiten und zum Erstellen von Texten. Er kann als News-Reader (s. Abschnitt 4.3) eingesetzt werden und sogar als WWW-Browser. Außerdem ist er ein beliebter EMail-Client. Hier soll eine kurze Einführung gegeben werden, um die Anfangsschwierigkeiten zu vermeiden.

Woher die Bezeichnung Emacs kommt, ist nicht ganz klar. Verschiedene mehr oder weniger humorvolle Varianten kursieren. „Eight Megabytes of Memory Continuously Swapping“ soll als Hinweis verstanden werden, daß Emacs nicht sehr sparsam mit dem Speicherplatz umgeht. Das sollte – wenn der Rechner nicht ohnehin völlig überlastet ist (z. B. durch Netscape) – keine Rolle mehr spielen. „Escape-Meta-Alt-Control-Shift“ ist die Andeutung, daß beim Emacs Kommandos wie *Laden*, *Speichern*, *Suchen und Ersetzen* usw. mit Tastenkombinationen *control*+`X`+`F`, *control*+`X`+`S`, *meta*+*shift*+`%` usw. eingegeben werden. Sicher ist es hilfreich, einige dieser Konbinationen im Kopf zu haben, neuere Versionen des Emacs sind aber maus- und menüorientiert und daher recht einfach zu bedienen.

Es gibt zwei Varianten des Emacs, den „normalen“ Emacs und den XEmacs. Letzterer ist eher graphisch orientiert, mit kleinen Bildchen, wie man sie von Windows-Rechnern gewohnt ist. Man startet XEmacs durch Eingabe des Befehls

```
xemacs &
```

(Das `&` bewirkt, daß XEmacs im Hintergrund gestartet wird, die Shell also wieder zur Verfügung steht.)

Vor dem Start sollte man sich die Konfigurationsdatei `.emacs` (s. Abschnitt 6.2) aus `~faensen/lehre/brueckenkurs/.emacs` in sein Heimatverzeichnis kopieren:

```
cp ~faensen/lehre/brueckenkurs/.emacs ~
```

Die herausragenden Features von Emacs/XEmacs sind

- Vielseitigkeit durch seine Einsetzbarkeit als Programmentwicklungsumgebung, EMail-Client, News-Reader ...
- Syntax-Highlighting *(font lock)*, d. h. farbiges Hervorheben von Schlüsselwörtern zur Unterstützung bei der Programmentwicklung,
- einen Haskell-Modus[13], der für die Teilnehmer am Kurs „Algorithmen und Programmierung I“ interessant sein dürfte.

[13] `http://www.cs.york.ac.uk/~gem/haskell-mode/`

6.1 Die wichtigsten Tastenkombinationen und Kommandos

Die folgende Tabelle faßt die wichtigsten Tastenkombinationen zusammen. Die Notation der Tastenkombination entspricht der, die in der Emacs-Dokumentation üblich ist. Hier wird *control* mit `C`, *meta* (das ist die Taste mit der Raute direkt neben der Leertaste) mit `M` und *shift* mit `S` bezeichnet. `C-x` bedeutet, daß gleichzeitig *control* und `x` gedrückt werden sollen. Groß- und Kleinschreibung wird unterschieden, d. h. `M-%` verlangt gleichzeitiges Drücken von *meta*, *shift* und der `5`.

Tastenkombination	Bedeutung
`C-x C-c`	Emacs beenden
`C-x C-f`	Datei öffnen
`C-x C-s`	Datei speichern
`C-x C-w`	Datei unter anderem Namen speichern
`C-x 2`	Zwei Fenster geöffnet
`C-x 1`	Ein Fenster geöffnet
`C-x b`	Puffer wechseln
`C-k`	Löschen bis Zeilenende (in Zwischenablage)
`C-y`	Einfügen aus Zwischenablage
`C-g`	Kommando abbrechen
`ESC-ESC-ESC`	Kommando abbrechen
`C-h k`	Beschreibe die folgende Tastenkombination
`C-h b`	Liste Tastaturbelegung auf
`C-h i`	Emacs-Dokumentation
`C-s`	Suche vorwärts
`C-r`	Suche rückwärts
`M-%`	Suchen und Ersetzen (y=yes, n=no, !=Rest)
`TAB`	Einrücken oder vervollständigen
`[f5]`	Gehe zu Zeile (für die Programmierung unentbehrlich)

Die wichtigsten Kommandos erreicht man auch über das Menü. Andere lassen sich nur über die Tastatur eingeben. Dazu drückt man `M-x` und gibt das entsprechende Kommando ein. Wichtige Kommandos sind

Kommando	Bedeutung
`auto-fill-mode`	Automatischer Zeilenumbruch
`goto-line`	Gehe zu Zeile x
`recover-file`	Wiederherstellen nach Absturz

Der Emacs hat mehrere Puffer *(buffer)*, einen für jede Datei, die aktuell geöffnet ist. Möchte man mehrere Dateien parallel editieren, so sollte man also keinesfalls Emacs erneut starten. Man belastet sonst unnötig den Arbeitsspeicher, was sich unschön auf die Geschwindigkeit des Rechners auswirkt. Man wechselt zwischen Puffern mit `C-x b`, zwei Puffer gleichzeitig betrachten kann man mit `C-x 2`.

6.2 Die Konfigurationsdatei .emacs

Emacs wird konfiguriert über eine Dot-Datei im Heimatverzeichnis, `.emacs`. Die Konfiguration erfolgt in Emacs-Lisp, einer Programmiersprache, die gerade für Anfänger nicht leicht zu handhaben ist. Eine Muster-`.emacs`-Datei, die ein sofortiges Loslegen ermöglicht, wurde für die Kursteilnehmer erstellt und steht unter

`~faensen/lehre/brueckenkurs/.emacs`

zum Kopieren bereit. Sie ist so kommentiert, daß die Einstellungen verständlich sein sollten.

6.3 Dokumentation

Die Dokumentation des Emacs und seiner Module weicht ein wenig von den sonst üblichen Dokumentationen ab. Die *man pages* helfen hier nicht viel weiter. Das Hilfesystem des Emacs nennt sich *Info*. Man ruft es mit `C-h i` auf. Dazu muß aber die Umgebungsvariable `INFOPATH` auf /opt/SUNWspro/contrib/XEmacs20.0-b28/lib/xemacs-20.0-b28/info gesetzt sein.

7 Der Editor vi

Der `vi` ist unter Informatikern geliebt und gehaßt. Für die liebenden, die Puristen, gilt er als eines der mächtigsten Werkzeuge überhaupt, überall verfügbar, stabil wie sonst nichts, und wer ihn bedienen kann, gehört einfach dazu, jedenfalls in bestimmten Kreisen, in anderen wird man gemieden. Gehaßt wird er, weil seine Bedienung nicht erlernbar ist, ohne einen Pakt mit dem Teufel zu schließen. Geschrieben im Jahre 1973, also schon fast dreißig Jahre alt, fehlt dem `vi` jede graphische Benutzerführung. Menüs oder gar die Maus sind ihm unbekannt. Jede Eingabe erfolgt über Tastenkombinationen, die so kryptisch sind, daß selbst der Erwerb von Grundkenntnissen malaiischer Sprachen wie ein Kinderspiel wirkt. Genug der Polemik. Woher kommt die Motivation, sich dennoch mit dem `vi` zu beschäftigen?

- Wie gesagt: `vi` ist überall verfügbar. Er gehört zur Unix-Standardausstattung. Das bedeutet, wenn kein anderer Editor installiert ist, wird man auf den `vi` ausweichen müssen. Insbesondere Systemadministratoren kommen daher um den `vi` nicht herum.

- `vi` ist wirklich sehr mächtig. Wer nicht vor monatelangem Studieren der zahllosen Einführungen zurückschreckt und mit viel Versuch und Irrtum diverse Arbeit zunichte gemacht hat, wird irgendwann in der Lage sein, mit einem einzigen Befehl die komplexesten Änderungen in seinen Programmquellen durchzuführen. (Man darf aber in der Zeit des Wissenserwerb um den `vi` nichts anderes tun, insbesondere keine anderen Programme nutzen oder mit Menschen reden, sonst kommt es zu einem Phänomen, das ich mal als kognitive Interferenz bezeichnen möchte.)

- Wenn man statt `vi` `vim` *(vi improved)* benutzt, hat man etwas mehr Komfort.

- Hauptargument für die Auseinandersetzung mit `vi` ist, daß es einem leicht passieren kann, ihn auf einmal gestartet zu haben und nicht mehr herauszufinden. Das liegt daran, daß manche Programme standardmäßig `vi` als Texteditor verwenden. Dazu gehören `elm`, `more` und `less`. Durch Setzen der Umgebungsvariablen `EDITOR` läßt sich dies manchmal ändern.

Das wichtigste zum Umgang mit `vi` ist zu wissen, wie man ihn beendet, und zwar sowohl mit als auch ohne Speichern etwaiger Änderungen in einer Datei. Zunächst aber zum Aufruf:

```
vi filename(s)
```

Mit `vi +`*zeilennummer* `filename` wird der Cursor gleich in der Zeile *zeilennummer* positioniert. Statt `vi` kann natürlich auch `vim` aufgerufen werden.

Der `vi` hat zwei verschiedenen Modi, den *Kommando*modus und den *Einfüge*modus. Nach dem Start befindet man sich im Kommandomodus. Was hier eingegeben wird, wird als Kommando interpretiert, nicht als Text. Das ist sicher sehr ungewohnt. In den meisten Editoren kann man direkt mit der Eingabe von Text loslegen. Mit `i` *(insert)* wechselt man in den Einfügemodus. Jetzt kann man schreiben, aber z. B. nicht mit den Cursortasten im Text herumwandern. Zurück in den Kommandomodus kommt man mit `ESC`. Die wichtigsten Kommandos sind:

Kommando	**Wirkung**
`i`	Wechsel in Einfügemodus
`o`	Zeile unten einfügen
`O`	Zeile oben einfügen
`:w`	Speichern der Datei
`:q`	Verlassen des `vi` (nur nach Speichern)
`:wq`	Speichern und Verlassen des `vi`
`:q!`	Verlassen ohne zu speichern
`:n`	Nächste Datei
`dd`	Zeile löschen
`D`	Rest der Zeile löschen
`dw`	Wort rechts löschen
`db`	Wort links löschen
`u`	Rückgängig
`/`*such*	Suchen nach *such*
`n`	Wiederhole letztes Suchen
`1,$s/`*xxx*`/`*yyy*`/g`	Ersetze im gesamten Text *xxx* durch *yyy*
n`G`	Gehe zu Zeile *n*

Da diese Einführung alles andere als erschöpfend ist, hier noch einige Verweise zu vi-Einführungen und -Referenzen:

- vi Tutorial[14]
- Eine vi Reference Card[15] (lokal als Postscript zum Ausdrucken)

[14] `http://ECN.www.ecn.purdue.edu/ECN/Documents/VI/`
[15] `vi-ref.ps`

- Die vi Lovers Home Page[16] mit vielen Verweisen zu anderen Quellen/Handbüchern …

8 Etikette

Wichtige Informationen zum Umgang mit seinen Mitmenschen bzw. -benutzern und den Ressourcen des Instituts finden sich im WWW[17]. Einige Punkte seien hier noch einmal zusammengefaßt.

- Einen Arbeitsplatz über längere Zeit (z. B. Mensagang) mit `xlock` o. ä. zu blockieren, ist unfreundlich und in Zeiten, in denen Rechner knapp sind, tabu. Längere Prozesse laufen auch nach dem Ausloggen weiter, wenn sie mit `nohup` gestartet werden. Bsp.: Statt

  ```
  analyze huge_mount_of_data
  ```

  benutze

  ```
  nohup analyze huge_mount_of_data
  ```

- Sehr rechenintensive Programme kann man so aufrufen, daß sie anderen Programmen eher den Vortritt lassen. Sie laufen dann insgesamt langsamer, die weitere (vor allem interaktive) Arbeit am Rechner wird aber nicht so ausgebremst. Hierzu dient das Kommando `nice`. Bsp.:

  ```
  nice rechenintensives_programm
  ```

- Wer nur ein älteres Modell einer Workstation ergattert hat, ist in der Versuchung, große, sonst sehr langsame Programme wie Netscape auf einem schnelleren Rechner laufen zu lassen. Das funktioniert, wie in Abschnitt 3.14 beschrieben. Natürlich sind dabei einige Kleinigkeiten zu beachten. Zunächst vergewissere man sich, daß auf dem Rechner Ressourcen frei sind. `rusers -i` listet alle Rechner im lokalen Netz sortiert nach freier CPU-Zeit auf. Nicht jeder Rechner ist aber öffentlich, auch wenn ein *remote login* möglich ist. Insbesondere von einer Belastung der Arbeitsplatzrechner der Wissenschaftler ist abzusehen. Kommt hinzu, daß man diese mit fehlerhaft programmierter freier Software in die Knie zwingt, handelt man sich Ärger ein.

[16] `http://www.cs.vu.nl/~tmgil/vi.html`
[17] `http://www.inf.fu-berlin.de/tec/etikette.html`

- Die Heimatverzeichnisse anderer Benutzer sind häufig lesbar. Herumschnüffeln darin ist unfein.

- Hat jemand vergessen, sich auszuloggen, so sollte der folgende Benutzer an diesem Rechner dies still für ihn erledigen, ihn vielleicht noch freundlich daran erinnern, das nächste mal selbst daran zu denken. Mißbrauch der Lage zerstörerischer (z. B. Dateien ändern oder löschen) oder auch nur „lustiger“ Art (z. B. versenden von EMails unter dem Namen des Opfers) grenzt an Kriminalität.

- Wer beim Versuch, Paßwörter der Benutzer zu erspähen, erwischt wird, verliert mindestens seinen Account.

- Es gibt am Institut verschiedene Mail-Verteiler. Wird an eine solche Adresse eine EMail geschickt, so erreicht man damit alle Mitglieder des Verteilers. Mit einer Mail an `tut` erreicht man beispielsweise alle Tutoren, über `stud` alle Studenten. Ein beliebtes Fettnäpfchen für Erstsemesterstudenten ist der Verteiler `all`. Wer an diesen eine Mail adressiert, nervt nicht nur die anderen Studenten, sondern auch alle Mitarbeiter und Professoren des Instituts. Die Benutzung dieses Verteilers ist der Technik vorbehalten. Mitteilungen, die für alle von Interesse sind, gehören in die News (4.3).

  Wichtig ist vor allem der Verteiler `staff@inf.fu-berlin.de`. Hierüber erreicht man die Technik, was insbesondere für Problemberichte nötig ist. Äußerungen von Wünschen wie „Hier ist `gcc` nur in der bereits seit 14 Tagen veralteten Version 2.8.0 installiert, ich verlange ein Update auf 2.8.0a“ werden in der Regel bestenfalls ignoriert.

- Spiele sind erlaubt. Man frage sich aber, ob man nichts besseres zu tun hat. Außerdem ist es unschön, damit einen Rechner zu blockieren, wenn andere darauf warten, endlich ihre Übungsaufgaben lösen zu können.

- Essen und Trinken in den Rechnerräumen ist verboten, Rauchen im gesamten Institut mit Ausnahme der Innenhöfe.

9 Sicherheit im Netz (von Kristine Budde)

Niemand würde angesichts der Kriminalitätsstatistiken seine Haustür sperrangelweit offenstehen lassen oder sich angesichts des allgemeinen Verkehrsaufkommens ohne Führerschein in ein Auto setzen und zur Rush Hour einfach drauflosfahren und dabei ernsthaft der Meinung sein, es würde schon nichts passieren.
Genauso sollte man sich auch im neuenMedium der vernetzten Computer über mögliche Risiken informieren, und ggf. Sicherheitsvorkehrungen treffen. Einige Risiken werden in diesem Abschnitt beschrieben und Lösungen sollen genannt werden.

9.1 Internetsecurity

- **Cookies**
  Cookies sind kleine Textsequenzen, die beim Besuch vieler Web-Seiten (z.B. AltaVista) in einer bestimmten Datei (z.B. in /.netscape/cookies) abgelegt werden. Diese protokollieren i.d. Regel, welche Seite einer Site besucht wurden. Damit kann später gezielt Werbung eingeblendet oder Werbe-EMail verschickt werden.

  In den meisten Browsern kann man entweder verlangen, daß man vor dem Setzen eines Cookies gefragt wird, oder das Setzen der Cookies ganz verbieten.

- **JavaScript**
  JavaScript ist inzwischen weit verbreitet, um Web-Seiten bunter und dynamischer zu gestalten. Allerdings kann es auch dazu mißbraucht werden, Profile von Web-Benutzern zu erstellen, um so noch zielgerichteter Werbung zu verschicken. Abgesehen davon ist JavaScript immer noch so fehlerhaft programmiert, daß damit praktisch beliebiger Schaden angerichtet werden kann.

  Auch JavaScript kann bei den meisten Browsern ausgeschaltet werden.

- **Übertragung sensitiver Daten über das Internet**
  Sicherlich wird man beim Surfen durch das Internet oft auf Seiten treffen, von denen man sich weiteres Informationsmaterial, z.B. auch in schriftlicher Form wünscht. Dabei sollte man allerdings beachten, daß die angegeben Daten in der Regel unverschlüsselt übertragen werden. Spätestens, wenn eine Bestellung mit VISA-Karte bezahlt werden soll, sollte man darauf achten, daß das Bestellformular als sicher gekennzeichnet ist, z.B. mit Hilfe von SSL oder S-HTTP. Das erkennt man z. B. durch die Benutzung von https://... oder shttp://... bei der Web-Adresse.

Interessantes zum Thema Internetsecurity findet man z.B. auf der Homepage von Richard Kemmerer[18].

9.2 ftp

ftp ist sicherlich die einfachste Möglichkeit, Daten von einem Ort zum anderen zu übertragen. Allerdings sollte man dabei bedenken, daß bei einem nicht-anonymen ftp das Paßwort unverschlüsselt über die Leitung geschickt wird. Stattdessen bietet sich die Benutzung eines Programms namens scp (Secure Copy) an, das es überall dort gibt, wo auch ssh (s. nächster Abschnitt) installiert ist. Benutzung:

```
scp <loginname>@<rechnername>:<quelldatei_mit_pfad> <loginname>@<rechnername>:<zieldatei_mit_pfad>
```

[18] `http://www.cs.ucsb.edu/~kemm/`

9.3 ssh

Das Programm ssh (Secure SHell) dient dazu, sich auf einem anderen Rechner sicher einzuloggen. Sicher meint dabei, daß sämtliche Kommunikation zwischen den Rechnern verschlüsselt wird. Dies kann auch dazu genutzt werden, wie in Kap. 3.13 beschrieben, remote Programme aufzurufen, deren Kommunikation dann verschlüsselt wird. Benutzung:

```
ssh -l <benutzername> <rechnername>
```

9.4 EMail als virtuelle Postkarte → pgp

Jeder der EMail benutzt, weiß sicher ihre Vorteile wie die schnelle und kostengünstige Zustellung zu schätzen.

Allerdings sollte man beachten, daß EMails nichts anderes sind als virtuelle Postkarten, man sollte es daher tunlichst vermeiden, sensitive Daten wie z.B. Paßwörter per EMail zu versenden, da EMails von allen Knoten, die sie auf ihrem Weg durch das weltweite Netz passieren, ohne Probleme gelesen werden können.

Um dem abzuhelfen, gibt es ein Programm namens pgp. Dies kann dazu benutzt werden, Nachrichten so zu verschlüsseln, daß sie nur der Empfänger lesen kann oder sie so zu signieren, daß leicht geprüft werden kann, ob die Nachricht tatsächlich vom Absender stammt.

Benutzung: (folgt)

9.5 Paßwortkriterien

Folgende Konventionen sollten bei der Wahl eines Paßwortes beachtet werden:

- Es dürfen keine Wörter aus **irgendeinem** Sprachgebrauch benutzt werden. Alle Wörter, die in irgendeiner Sprache (und sei es Mandarin) irgendeine Bedeutung haben, sind mit Sicherheit auch in einem elektronischen Wörterbuch gespeichert, das dazu genutzt werden kann, sog. Klartextangriffe auf das Paßwort zu starten. Dies gilt auch für beliebte Namen aus der Literatur, so existiert z.B. ein Tolkien Wörterbuch! Es reicht auch nicht aus, bestimmte Buchstaben dieser Wörter durch Zahlen oder Sonderzeichen zu ersetzen (z.B. l durch 1 oder O durch 0)!!!
- Das Paßwort sollte 7 oder 8 Buchstaben lang sein, keinesfalls kürzer.
- Es sollten Groß- und Kleinbuchstaben mit Sonderzeichen und Zahlen vermischt benutzt werden.
- Das Paßwort darf nirgendwo notiert werden.

Gute Paßwörter sind zu schwer, um leicht geknackt zu werden, aber leicht genug, daß man sie sich merken kann.

Am Besten bildet man Paßwörter anhand bestimmter Schemata, die man sich leicht merken kann (z.B. zwei kurze Wörter mit Sonderzeichen abwechselnd verknüpft oder den ersten, zweiten, dritten Buchstaben des ersten, zweiten, dritten Wortes eines lustigen Satzes benutzen, wobei man bestimmte Buchstaben durch Zahlen oder Sonderzeichen ersetzt).

9.6 Gibt es einen EMail Virus - Viren unter UNIX

Im Gegensatz zu DOS oder Windows wird man unter UNIX keine echten Viren antreffen.

Auch das Gerücht von Viren, die durch das Lesen von EMail Schaden anrichten können, ist völliger Blödsinn. Problematisch ist es allerdings, wenn man seine Mail mit einem Mail-Programm liest, das eigenstndig in der Lage ist, in der Mail enthaltenen Programmcode zur erkennen, und diesen dann auch ausführt, wie z.B. bei netscape-Mail.

Wenn man allerdings freiwillig ein Programm, daß einem von irgendjemanden per EMail zugeschickt wird, auf dem Rechner laufen lässt, dann ist das kein Virus, der da ggf. Schaden angerichtet hat, sondern die eigene Dummheit. Sie würden ja schließlich auch sonst kein Programm auf ihrem Rechner laufen lassen, daß ihnen irgendein Fremder auf der Straße in Diskettenform in die Hand gedrückt hat.

Allgemein sollte man Software aus dem Internet möglichst nur von offiziellen Distributoren beziehen.

10 Windows NT

Das Institut hat auch einige PCs, die unter Windows NT laufen. Studenten haben dort Zugang zu gängiger Büro-Software wie MS Word, MS Excel usw. Für die Teilnehmer am Kurs „Algorithmen und Programmierung I" ist zu betonen, daß die dort verwendete Programmiersprache Haskell unter Windows eine besonders komfortable Bedienungsumgebung hat. Den Kursteilnehmern wird freigestellt sein, ob sie mit der Windows- oder der Unix-Version arbeiten möchten.

Den meisten wird die Arbeit mit Windows vertraut sein. Eine Einführung wird daher nicht gegeben. Wichtig ist nur, daß auf den Windows NT-Rechnern das gleiche Paßwort gilt wie auf den Unix-Rechnern. Auch kann auf das eigene Heimatverzeichnis zugegriffen werden. Es wird unter Windows wie ein Laufwerk angesprochen (i. d. R. unter dem Laufwerksbuchstaben Z:).

Übrigens gibt es auch unter Windows NT Telnet. Man kann sich von hier aus auf den Sun Workstations einloggen, aber natürlich keine Programme mit graphischer Oberfläche benutzen. Sollten alle Suns belegt sein, kann man von hier aus aber wenigstens seine EMail lesen.

11 Troubleshooting

- **Ein Programm reagiert nicht mehr**
  Mit

  ```
  jobs -l
  ```

  erhält man eine Liste aller laufenden Programme. In der zweiten Spalte der Liste steht eine Identifikationsnummer des Programms:

  ```
  [2]   5482 Running      netscape &
  [3]-  5751 Running      xemacs  &
  [4]+  8074 Running      perfmeter &
  ```

  Angenommen, das Programm `perfmeter` macht Probleme und läßt sich nicht mehr ordentlich beenden. Ein Ende erzwingen kann man mit

  ```
  kill 8074
  ```

  `8074` ist hierbei die Identifikationsnummer des zu beendenden Programms. Funktioniert dies nicht, so hilft garantiert der Switch `-9`

  ```
  kill -9 8074
  ```

  Fensterorientierte Programme lassen sich auch mit `xkill` beenden. Der Mauszeiger ändert sich, und man kann das Fenster anklicken, dessen Programm man beenden möchte.

- **Eine Datei namens `core` liegt herum**
  Die kann man getrost löschen

- **Ich kann mich nicht einloggen**
  Eventuell sind die Platten-Quotas erschöpft (s. Abschnitt 3.10). Oder das Paßwort war zu einfach und wurde geknackt. Bei einem „normalen“ Kommandozeilen-Login wird man informiert, warum das einloggen verweigert wird. Dies geschieht bei der CDE leider nicht. Versuche, im Login-Fenster statt CDE einen Kommandozeilen-Login zu erzwingen, um näheres zu erfahren.

- **Der Bildschirm ist schwarz, der Rechner aber an**
  Maus bewegen, Taste drücken, Bildschirm einschalten.

- **Wen soll ich fragen?**
  Zunächst den, der am Rechner nebenan sitzt. Dann andere Kommilitonen. Meist sitzen auch Tutoren im Tutorenraum im Keller. Wenn keiner sonst helfen kann, stehen die Mitarbeiter der Technik im ersten Stock für Fragen zur Verfügung.

12 Anhang

12.1 Weitere nützliche Kommandos

Die folgende Auflistung hat keinen Anspruch auf Vollständigkeit, soll die Funktion der Kommandos nicht erschöpfend behandeln und hat einzig den Zweck, Anregung zu geben, selbst in der Dokumentation nachzuschlagen.

grep Ein Programm zum suchen von Mustern in Textdateien (auch Programmquelltexten). Bsp.:

```
grep -i schmidt adressen.txt
```

liefert alle Zeilen in der Datei `adressen.txt`, die den Namen Schmidt (`-i`: unhabhängig von Groß-/Kleinschriebung) enthalten.

```
grep -n counter *.c
```

liefert alle Zeilen (`-n`: mit Zeilennummer) aller C-Programme im Verzeichnis (`*.c`), in denen `counter` auftaucht.

```
grep -v x file
```

liefert alle Zeilen der Datei `file`, in denen kein (`-v`) x auftaucht.

wc zählt Zeichen, Wörter und Zeilen. Bsp.:

```
wc < diplomarbeit.txt
```

head, tail liefern die ersten bzw. letzten Zeilen.

sort sortiert zeilenweise

```
sort < adressen.txt > adressen-sortiert.txt
```

talk Telephon für Taubstumme. Tastatureingaben erscheinen auf dem Bildschirm des „Gesprächspartners“. Anwendung: user1 sitzt am Rechner scholle und möchte mit user2, der am Rechner hai sitzt, „sprechen“

```
talk user2@hai
```

user2 erhält eine Nachricht, daß user1 mit ihm sprechen möchte und am Rechner scholle sitzt. Er antwortet mit

```
talk user1@scholle
```

`talk` wird mit *control*-C beendet.

finger Mit `finger` erfährt man, ob ein Benutzer an einer anderen Maschine eingeloggt ist:

```
finger schmidt@hai
```

ftp `ftp` dient dem Kopieren von Dateien zwischen verschiedenen Rechnern. Es ist insbesondere von Bedeutung für den Download frei verfügbarer Software *(anonymous ftp)*, sofern dieser nicht bereits über das WWW erfolgt. Innerhalb des lokalen Netzes wird ftp nicht benötigt.

Index