Projektgruppe Praktische Mathematik
Technische Universität Berlin
Sekretariat MA 4-3

PPM-Intro

UNIX-Pool Einführung

Projektgruppe Praktische Mathematik *

11. Oktober 2002

* Quellen: `~ppm-soft/skripte/Unix-Skript/`
Zuletzt bearbeitet: Alexander Breitholz (11. Oktober 2002)

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Der UNIX-Pool

Der UNIX-Pool (Raum MA 241) ist eine Einrichtung des Fachbereichs Mathematik. Er steht den Studenten der hier stattfindenden Lehrveranstaltungen zur Bearbeitung ihrer Übungsaufgaben zur Verfügung.

Viele nützliche Informationen findet man unter `http://www-pool.math.tu-berlin.de/`

Im UNIX-Pool gibt es zwei verschiedene Arten von Rechnern. Einmal PowerPC mit AIX-UNIX als Betriebssystem und handelsübliche PC mit Debian Linux. An welchen Computern gearbeitet richtet sich nachdem welche Lehrveranstaltung Ihr besucht.

Achtung !! Bitte lest Euch die folgenden Punkte genau durch, wenn ihr sie beachtet, macht ihr uns und vor allem Euch die Arbeit leichter.

1.1 Ich melde mich an

Als erstes, bevor Ihr an die Rechner geht, müßt Ihr Euch oben auf der Empore anmelden. Wir brauchen dafür nur Euren Bibliotheksausweis. Anhand diesem sehen wir zu welcher Gruppe Ihr gehört. Ihr bekommt nun von den Unix-TutorInnen einen Schließfachschlüssel. Es dürfen keine Jacken, Taschen und Essenssachen oder Getränke mit an die Rechner genommen werden, also schließt diese Dinge bitte in den Schließfächern ein.

1.2 Wo ist mein Computer ?

Ihr findet Euren Computer anhand der Nummer auf eurem Schlüssel. Die Computer mit einer Nummer kleiner als 80 (Linux PC) befinden sich unten in dem Eingangsraum, alle Computer mit einer Nummer größer gleich 80 (IBM RS6000) befinden sich in dem Raum neben der Anmeldung mit der blauen Tür.

1.3 Woher weiß der Computer wer ich bin ?

Das Erste was Ihr seht, wenn Ihr an die Computer kommt, ist die unten abgebildete sogenannte Login-Maske. Sie beinhaltet zwei Eingabefelder für Euren Namen und euer Passwort. Als Login-Namen gebt Ihr das Kürzel eurer Lehrveranstaltung gefolgt von einem Bindestrich (auf der Tastatur ein Minus) und Eurer dreistelligen Gruppennummer ein. Danach bestätigt Ihr mit der RETURN-Taste, das ist die größte Taste auf der Tastatur, rechts außen.

Die Kürzel für Eure Lehrveranstaltung lauten wie folgt:

Lehrveranstaltung	**Login-Kürzel**
Informationstechnik/C	itc
Informationstechnik/Pascal	itp
Informationstechnik/Fortran95	itf
Projekt Praktische Mathematik	ppm

In dem Passwortfeld müßt Ihr Euer Passwort eintippen. Ihr habt beim ersten Mal noch kein Passwort, also laßt das Feld leer und drückt einfach nur die RETURN-Taste. Ein Beispiel:

```
Login: itc-107

Password:
```

Jetzt sollte die Login-Maske verschwinden und auf eurem Bildschirm zwei Fenster, eine Uhr und ein Briefkasten erscheinen.

1.4 Ich trage mein Passwort ein

Das Erste was Ihr tun solltet,damit niemand in Eurem Bereich herumpfuscht, ist ein Passwort für Euren Bereich einzutragen. Denkt Euch ein Passwort aus. Dabei solltet Ihr beachten, daß normale Wörter wie sie im Wörterbuch stehen, Euer Vor- oder Nachname (in jeglicher Kombination), Euer Geburtsdatum wie auch sonstige einfache Namen keine guten Passwörter sind. Das Passwort muß mindestens 6 Zeichen lang sein, es wird zwischen Groß- und Kleinschreibung unterschieden und es darf bzw. soll auch Sonderzeichen enthalten.

Übung: `password` **ändern.**
Um das Passwort zu ändern, gebt Ihr in einem der beiden Fenster den Befehl:

```
yppasswd
```

RETURN ein.
Es erscheint nun:

```
Old NIS password:
```

Hier gebt Ihr einfach nur RETURN ein, Ihr habt ja noch kein Passwort.
Daraufhin meldet der Computer:

```
New password:
```

Jetzt tippt Ihr das Wort ein, daß Ihr Euch ausgedacht habt und bestätigt am Ende mit RETURN. Ihr könnt auf dem Bildschirm nicht sehen, was Ihr tippt, mit dem Vorteil, daß es Euer Nachbar auch nicht kann.

```
Enter the new password again:
```

Um eventuelle Tippfehler auszuschließen, müßt Ihr hier das Passwort noch ein zweites Mal eingeben.
Wenn Ihr alles richtig gemacht habt, müßte am Ende so etwas wie:

```
NIS passwd changed on fb3-s1
```

Bitte sagt allen aus eurer Gruppe (nicht Lehrveranstaltung) wie das Passwort lautet, damit sich auch jeder einloggen kann, wenn er mal alleine kommt.

1.5 Wenn ich fertig bin und nach Hause gehen will

Wichtig !!! Auf keinen Fall den Computer ausschalten ! Wenn Ihr die Computer einfach abschaltet, kann es zu Datenverlusten kommen oder Hardware beschädigt werden.

Also nicht ausschalten, sondern ausloggen ($\cong$ abmelden). Um sich auszuloggen müßt Ihr in dem Fenster mit dem schwarzen Hintergrund den Befehl:

```
logout
```

RETURN eingeben.

Eure Fenster sollten nun alle vom Bildschirm verschwinden, und nach einer Weile erscheint wieder die Login-Maske für den nächsten Benutzer.
Wenn Ihr Euch nicht ausloggt, hat der Nächste, der an Euren Rechner kommt, freien Zugriff auf all Eure Daten. Also er könnte sie auch alle löschen, wenn er nur will.
Wenn Ihr Euch ausgeloggt habt, holt ggf. eure Sachen aus dem Schließfach und gebt den Schlüssel mit dem Hefter vorne an der Anmeldung wieder ab.

Übung: Ausloggen und wieder einloggen.
Natürlich ohne zwischendrin den Schlüssel abzugeben!

1.6 Ich habe Probleme, wen kann ich fragen ?

Grundsätzlich gibt es im Leben zwei Arten von Problemen:

1. Probleme mit den Aufgaben eurer Lehrveranstaltung.
In diesem Fall solltet Ihr erst das entsprechende Kapitel in eurem Skript durcharbeiten und **dann** eure Assistenten/Tutoren um Rat fragen.

2. Probleme mit den Rechnern.
Bitte **niemals** zur Selbstjustiz mit den Rechner schreiten, d.h. auf keinen Fall irgendwelche Knöpfe, die sich nicht auf der Tastatur befinden, wie z.B. Reset, Power oder ähnliches drücken. Wendet Euch an die Unix-TutorInnen, das sind die Leute, die auf der Empore an der Anmeldung sitzen.

2 UNIX - Ein Betriebssystem ?!

Ein Betriebssystem ist sozusagen das Verwaltungsprogramm des Computers. Es steuert und vermittelt zwischen den einzelnen internen als auch externen Geräten, wie z.B. Festplatte, Monitor, Drucker, Speicher, Prozessor, Netzwerk ....
Die Rechner an denen Ihr arbeitet, laufen unter dem Betriebssystem namens UNIX. Andere mehr oder weniger vorbildliche Beispiele für Betriebssysteme sind Linux, Windows98, Windows2000, Windows XP, Apples Mac OS X oder das gute (sehr) alte MS-DOS.

UNIX hat seine Verbreitung neben Industrie und Forschung in erster Linie im universitären Bereich. Es ist ein echtes “Multitasking-Multiuser” Betriebsystem, d.h. auf einem Rechner können mehrere Benutzer mehrere Programme zur gleichen Zeit ablaufen lassen[1].

Ein weiterer Schwerpunkt liegt in den Netzwerkfähigkeiten von UNIX. Die Entwicklung ging Anfang der 70’er Jahre vom amerikanischen Militär aus. Ziel war es, ein Informationssystem bestehend aus einem Netzwerk von Computern zu errrichten, in dem die Daten selbst bei einem Ausfall einzelner Netzteile (sprich Kriegsfall) immer noch sicher Ihr Ziel erreichen. Diese Netzwerktechnik wurde in das Betriebssystem UNIX integriert und in der Praxis angewandt.

Nach und nach wurden neben militärischen Einrichtungen auch Universitäten und Forschungsinstitute in dieses Netzwerk eingebunden. Die Entwicklung nahm Ihren Lauf und brachte das umfangreichste Computer-Netzwerk der Welt - das **Internet** hervor.

Die Rechner an denen Ihr sitzt, sind ein Teil davon, auch wenn es nicht den Anschein macht.

[1] ohne das ein unbehebbarer Fehler im Anwendungsprogramm auftritt

3 Die Benutzeroberfläche

Als Teilnehmer einer Lehrveranstaltung haben wir Euch einen Account (Bereich) eingerichtet. Egal an welchem Rechner Ihr Euch einloggt, Ihr findet immer denselben Bereich vor. In Eurem Bereich habt Ihr alle Rechte und könnt nach Belieben neue Dateien und Ordner (Verzeichnisse) anlegen und wieder löschen.

Nachdem Ihr Euch eingeloggt habt, sollte Euer Bildschirm ungefähr wie folgt aussehen:

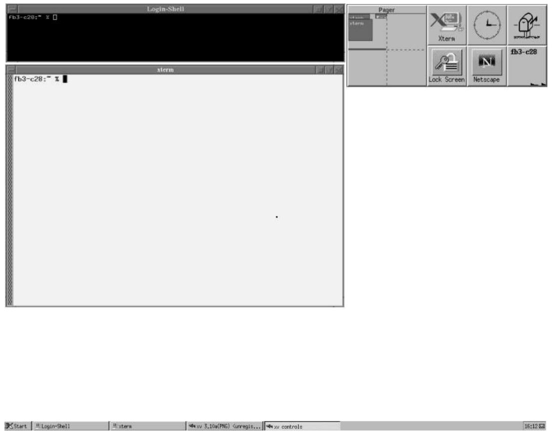

Abbildung 1: Desktop unter FVWM2

Diese Oberfläche nennt sich FVWM2 und ist ein Standard für fast alle UNIX-Systeme, d.h. solltet Ihr mal woanders an einem UNIX-Rechnern arbeiten, werdet Ihr unter anderem auch FVWM2 vorfinden.

Die Uhr dient zu Eurer Information. Der Briefkasten zeigt Euch an, ob noch ungelesene Post (E-Mail)[2] in Eurer Mailbox ist, aber dazu später mehr.

Die beiden Fenster (das schwarze und das weiße) nennt man **X-Terminals** (kurz: Xterm). Man kann sie als eigenständige Terminals ansehen, über die Ihr dem Rechner Befehle geben könnt. Das schwarze Fenster solltet Ihr nur zum ausloggen benutzen. In dem weißen Fenster macht Ihr dann alles andere.

3.1 Fenster Verändern

Jedes Fenster ist mit einem Rahmen versehen. Geht man mit dem Maus-Zeiger auf den Rahmen und drückt die linke Maustaste, kann man das Fenster vergrößern oder verkleinern (die Taste solange gedrückt halten, bis das Fenster die gewünschte Größe hat). Jedes Fenster hat weiterhin im Rahmen eine Kopfzeile, in der der Name des Fensters, bzw. Informationen stehen. Durch Anklicken dieser Zeile könnt Ihr das komplette Fenster verschieben.
Rechts von der Kopfzeile sind zwei quadratische Felder, das eine mit einem Punkt, das andere mit

[2] Electronic-Mail

einem Quadrat. Durch Anklicken des Quadrates wird das Fenster bildschirmfüllend vergrößert. Durch Anklicken des Punktes wird das Fenster verkleinert –*ikonifiziert*– und am linken unteren Bildschirmrand abgelegt. Um dieses rückgängig zu machen, klickt man einfach das "Ikon" zweimal schnell hintereinander (Doppelklick) mit der rechten Maustaste an.

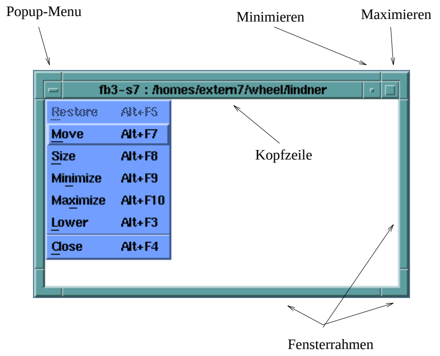

Abbildung 2: Fenster verändern

Ein Popup-Menü erscheint, wenn Ihr das kleine quadratische Feld mit einem Querbalken links neben der Kopfzeile anklickt. Hiermit kann man dann auch die Vergrößerung rückgängig machen. Das Menü enthält noch einmal verschiedene Funktionen, um das Fenster zu beeinflussen, am besten probiert es einfach mal aus. Mit CLOSE wird schließlich das Fenster geschlossen (Vorsicht mit diesem Befehl, denn Ihr könnt Euch z.B. im Editor nicht gesicherte Daten abschießen).

3.2 Die Maus

Wenn Ihr Eingaben in ein bestimmtes Fenster machen wollt, müßt Ihr dieses vorher mit der Maus aktivieren. Dazu geht Ihr mit dem Mauszeiger auf das entsprechende Fenster und drückt den linken Mausknopf. Bei dem aktiven Fenster ist der Rahmen dunkler als bei den anderen.

Der Umgang mit der Maus ist für Anfänger immer etwas problematisch, deswegen hier noch einige Tips: Der Rechner benötigt für das Ausführen von Anweisungen immer ein wenig Zeit, wenn Ihr also mit der Maus etwas angeklickt habt, und es passiert nicht gleich etwas, muß das nicht daran liegen, daß Ihr etwas falsch gemacht habt. Wartet also immer erst eine Weile. Auf keinen Fall solltet Ihr aber ein und die selbe Sache mehrmals anklicken. Es könnte Euch sonst z.B. passieren, daß Ihr statt einem, dreißig Xterm-Fenster auf einmal öffnet.

Eine sehr nützliche Funktion der Maus ist die Möglichkeit, Text zu markieren und diesen zu kopieren. Mit gedrückter linker Maustaste könnt Ihr Textteile markieren und mit der mittleren Maustaste an eine andere Stelle in ein beliebiges Fenster kopieren.

4 Crashkurs UNIX

4.1 Dateien und Verzeichnisse

In diesem Kapitel sind immer wieder kleinere Übungen auszuführen, die nicht extra gekennzeichnet sind.

Eine Datei (engl. file) ist eine Aneinanderreihung von Zeichen. Zunächst sollt Ihr eine kleine Datei schreiben. Dazu benötigt Ihr einen Editor.

Der Editor ist ein Programm, das einen dabei unterstützt, einen Text mit Hilfe des Computers zu schreiben und als Datei (engl. file) zu speichern. Dieses sollt Ihr jetzt mit Hilfe des Editors `xemacs` machen (oder `emacs`, `nedit`...):

Die einzelnen Schritte in Kurzform:

1. Ruft den Editor auf. Der Editor `xemacs` wird durch folgenden Befehl aufgerufen. Als Filenamen denkt Euch einen beliebigen Namen `FileName` aus, unter dem die Datei gesichert werden soll.

   ```
   xemacs FileName &
   ```

2. Wählt das Editorfenster mit der Maus an.
3. Schreibt einen Text im Arbeitsfeld des Editors.
4. Sichert den Text, der im Arbeitsfeld steht, auf ein File. Dazu drückt Ihr mit der linken Maustaste auf den Knopf FILE und haltet die Maustaste gedrückt. Jetzt geht Ihr in dem sich öffnenden Menü auf die Option SAVE FILENAME und laßt die Maustaste los.

So, jetzt habt Ihr Eure erste Datei auf dem Rechner geschrieben.

Als nächstes sollt Ihr die gleiche Datei unter einem anderen Namen abspeichern. Das macht Ihr, indem Ihr im Editor wieder auf FILE geht und auf dem Punkt SAVE AS die Maustaste loslaßt. Es erscheint ein kleines Fenster. Schreibt einen neuen Namen und klickt OK an. Jetzt schreibt noch einen anderen kurzen Text und speichert ihn unter einem dritten Namen ab.
Geht jetzt wieder mit der Maus in ein `xterm`-Fenster (aktivieren durch anklicken des Fensters) und gebt den Befehl `ls` ein. Der Befehl `ls` steht für "list", zu deutsch auflisten. Jetzt müßten die Namen Eurer Dateien aufgelistet sein. Gebt als nächstes `ls -l` ein. Jetzt erhaltet Ihr mehr Informationen über Eure Dateien. Macht Euch noch nicht zuviele Gedanken über die Bedeutung der Zeichen. Es müßte Euere Benutzernummer, das Datum der letzten Änderung und der Name der Datei zu sehen sein.

Als nächstes kopiert Ihr Euch eine Eurer Dateien auf den Namen 'kopie1'. Ihr macht das mit dem Befehl

```
cp FileName1 FileName2
```

`cp` steht für "copy" (deutsch: kopieren). Die erste Datei `FileName1` ist die Quelldatei, die zweite `FileName2` die Zieldatei (Also für FileName1 setzt Ihr einen Eurer Dateinamen ein, für FileName2 'kopie1'.

Laßt Euch wieder die existierenden Dateien auflisten (mit `ls` oder `ls -l`).

Gebt den Befehl `more kopie1` ein. Mit `more` kann man sich den Inhalt einer Datei anschauen, es besteht aber keine Möglichkeit die Datei zu verändern.

Wie Ihr seht lassen sich ziemlich schnell Dateien erzeugen. Um etwas Ordnung in die Menge der Daten zu bringen gibt es Unterverzeichnisse.

Stellt Euch vor, Ihr habt eine Unmenge von Akten voll mit Informationen. Um bestimmte Informationen in den vielen Akten leicht wiederzufinden, werdet Ihr sie nach bestimmten Kriterien ordnen und Akten mit gleichen Kriterien in Schubladen zusammenfassen. Sammeln sich in einer Schublade zuviele Akten an, werdet Ihr diese nach weiteren Merkmalen unterteilen und sogenannte Unterschubladen einrichten.

Genau nach diesem Prinzip funktioniert das Dateisystem des Computers. Wie eine große Schublade mit jeweils ein oder mehreren Unterschubladen ineinander verschachtelt.

Wenn man nun eine bestimmte Information sucht, macht man erst die große Schublade auf, dann eine der Unterschubladen u.s.w., bis man am Ende bei der entsprechenden Akte angekommen ist.

Den Weg zu der entsprechenden Schublade nennt man im Dateisystem des Computers: **Pfad** (engl. path), die Schubladen **Verzeichnisse** (engl. directory) und die Akten, **Dateien** (engl. files).

Zu einem vollständigen Dateinamen gehört eigentlich auch immer der komplette Pfad, nur wenn man sich schon in einem Unterverzeichnis befindet, daß ein Teil des Pfadnamens enthält, kann man hier Namensabkürzungen vornehmen.

Die folgende Abbildung stellt ungefähr einen Ausschnitt des Dateisystems auf diesen Rechnern dar:

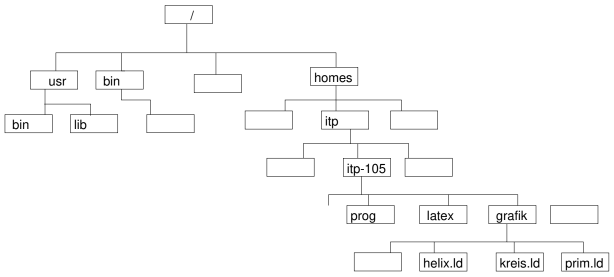

Abbildung 3: UNIX Dateisystem

Jedes Verzeichnis kann noch in weitere Unterverzeichnisse aufgeteilt sein.

In diesem Baum kann man natürlich auch von Ast zu Ast herumspazieren. Dies geschieht mit dem `cd`-Befehl (Change Directory).

Gebt zum Beispiel `cd /` an und listet Euch dann die verfügbaren Datei- bzw. Verzeichnisnamen auf (das `/`-Zeichen steht für die oberste Wurzel). Mit `cd Name` kommt Ihr in das entsprechende Unterverzeichnis. Mit `cd ..` kommt Ihr wieder ein Verzeichnis höher (zurück).

Ihr könnt in jedes Unterverzeichnis hineingucken, in dem Ihr ein Lesezugriffsrecht habt. Mit `more` könnt Ihr Euch Dateien anschauen. Wandert ein bißchen im Verzeichnisbaum herum und guckt Euch ein paar Dateien an (nehmt Euch dafür nicht mehr als ca. 10-15 min. Zeit). Wenn Ihr Euch verlaufen habt, kommt Ihr mit der Eingabe von `cd` ohne weiteren Namen automatisch nach Hause in Eure `home directory`. Mit dem Befehl `pwd` (Print Working Directory) könnt Ihr nachschauen, wo Ihr Euch gerade befindet (kompletter Pfad).

Nach dem Einloggen befinden wir uns automatisch in unserer **home directory**, z.B.:

```
/homes/itp/itp-123
```

Um später Ordnung in Euren Dateien zu haben, sollt Ihr Euch jetzt einige Unterverzeichnisse in Eurer `home directory` anlegen und zwar für folgende Bereiche jeweils ein Verzeichnis: Programmieraufgaben, Messen, Graphik, LaTeXund Schrott. Die Namen sollten so gebildet werden, daß sie nicht zu lang sind, aber klar ist, welche Themen in dem Verzeichnis enthalten sind. Achtung: Groß- und Kleinschreibung werden unterschieden.

Der Befehl zum Einrichten eines Verzeichnisses lautet `mkdir` (engl. make directory). Bsp.:

```
mkdir mess
```

Mit `cd` könnt Ihr natürlich auch wieder hin- und herwechseln.

Um ein Verzeichnis zu löschen verwendet man den Befehl `rmdir` abgeleitet von "remove directory". Man kann diesen Befehl natürlich nur ausführen, wenn man sich nicht in der zu löschenden Directory, sondern weiter oben oder auf einem anderen Zweig des Verzeichnisbaumes befindet (den Ast absägen auf dem man sitzt geht nicht)!

Löscht als nächstes das Verzeichnis für den Schrott.

```
rmdir schrottName
```

Natürlich kann man auch einzelne Dateien löschen. Dieses geschieht mit dem Befehl `rm FileName` (ReMove). Die folgende Abfrage beantwortet man mit `y` (für yes), wenn man sicher ist, daß die Datei gelöscht werden soll. Bei jeder anderen Eingabe wird das File nicht gelöscht!

Räumt mit `rm` in Eurem `home`-directory auf. Laßt ein File übrig, daß noch gedruckt werden kann. Zum Schluß laßt Euch mit `ls -l` auflisten, welche Verzeichnisse bzw. Dateien Ihr noch habt.

Das sind eigentlich schon die wichtigsten Unix-Befehle, die Ihr benötigt. Es gibt natürlich noch jede Menge mehr nützliche Dinge, doch dazu später mehr.

4.2 Überblick

In diesem Kapitel werden die wichtigsten Befehle des UNIX-Betriebssystems und die Namenbildung etwas ausführlicher vorgestellt. Dieses Kapitel kann quergelesen werden und dient eher zum nachschlagen. Achtung: Die Befehle `man` und `history` werden neu vorgestellt.

4.2.1 Datei, Verzeichnis, Pfad und Name

Eine Datei kann ein Programm, Zahlen oder z.B. einen Text enthalten. Auch der vorliegende Text ist als Datei auf der Festplatte gespeichert. Mit den Dateien kann man alles mögliche machen: Daten von einer Datei in eine andere übertragen; Dateien drucken; Dateien als Eingabe an ein anderes Programm übergeben usw.. Die wichtigsten dazu notwendigen Befehle, sollen nun nocheinmal erklärt werden.

Um eine Datei eindeutig zu kennzeichnen, braucht sie einen Namen. Komplette Namen von Dateien sind z.B.:

```
/homes/itp/itp-123/aufg1.p
```

Komplette File-Beschreibungen haben folgenden Aufbau:

```
Pfad/FileName.FileTyp
```

(dabei muß der '.' und der '/' mitgetippt werden, wobei der '.FileTyp' aber auch fehlen kann.)

Ein Pfad ist:

```
/Directory/Directory/Directory/...
```

Der File-Name

```
/homes/itp/itp-123/aufg1.p
```

ist also folgendermaßen zu zerlegen:

```
/homes/itp/itp-123/      aufg1.p
           Pfad          FileName.FileTyp
```

Oder zu deutsch: Es befindet sich

```
           im Pfad             /homes/itp/itp-123
           mit dem File-Namen  aufg1
           und dem Filetyp     p    (Pascal)
```

Bemerkung: Während man ein Pascal-Programm am Ende der File-Beschreibung mit `.p` kennzeichnet, benutzt man für ein C-Programm die Endung `.c` und für ein FORTRAN-Programm `.f` !

4.2.2 Wichtige UNIX-Befehle:

UNIX-Befehle werden in den Xterms nach dem **Prompt**

```
z.B.: fb3-c71<4>
```

eingegeben! Dieser Prompt ist eine Aufforderung an den Benutzer, einen Befehl einzugeben. Wenn der Prompt, also die Eingabeaufforderung, fehlt, nimmt der Computer keinen Befehl von Euch an.
Die Befehle bestehen grundsätzlich aus Buchstaben, Zahlen und manchmal noch weiteren Parametern dahinter.

Es sei vorher noch angemerkt, daß aus Gründen von Datenschutz und Systemsicherheit in UNIX verschiedene Rechte für die einzelnen Benutzer verwaltet werden. So ist genau festgelegt, was Ihr wo im Dateisystem dürft und was nicht, ob Ihr eine bestimmte Datei lesen, beschreiben oder ausführen dürft. So habt Ihr zum Beispiel nicht das Recht – moralisch wie auch computertechnisch die Aufgaben Euerer Nachbargruppe zu kopieren. Der Computer meldet Euch in solch einem Falle: `Permission denied`.

Der `ls`-Befehl

kommt von dem englischen "LiSt": Inhaltsverzeichnis auflisten (zeigen).
Man benötigt diesen Befehl, um die Namen aller Files/Unterverzeichnisse aufzulisten.

Die Eingabe des Befehls:

```
ls
```

listet alle Dateien in dem aktuellen Verzeichnis auf. Die Eingabe des Befehls mit dem Parameter `l` liefert mehr Informationen:

```
ls -l
```

Das Minuszeichen(-) steht in UNIX allgemein vor einem Parameter. Der Parameter `l` steht für long, also eine lange Information.
Mit dem Parameter `l` werden folgende zusätzliche Informationen ausgegeben:
Zuerst links stehen die Zugriffsrechte auf die einzelnenen Files, dann folgt eine Zahl, die die Anzahl der "links" (engl. Verbindungen) zu anderen Verzeichnissen und zu sich selbst angibt, danach kommt der Benutzername des `file`-Eigentümers gefolgt vom Gruppennamen, der Größe des `file`s in byte, dem Datum der letzten Änderung mit Uhrzeit und zuletzt der Name des `file`s. Die Zugriffsrechte bestehen jeweils aus 10 Zeichen:
An der ersten Stelle steht das Dateiartbit. Es kann folgende Einträge haben:

- `d` Datei ist ein Verzeichnis
- `l` Datei ist ein symbolischer Link, Soft-Link
- `-` Normale Textdatei
- `b` Blockorientierte Gerätedatei
- `c` Characterorientierte Gerätedatei

Als nächstes folgen 3 Stellen für die Zugriffsrechte des Benutzers (also Eure Zugriffsrechte), 3 Stellen für die Gruppe und schließlich 3 Stellen für alle anderen. Hierbei steht das 'r' für read (Leserecht), das 'w' für write (Schreibrecht) und 'x' für execute (Ausführrecht).

```
fb3-c32~: ls -l
drwx------    8 alex     ppm-b         106 Jul 11  2001 CVODE
-rw-------    1 alex     ppm-b        3575 May 24 10:53 readme.txt
lrwxrwxrwx    1 alex     ppm-b          12 Jul 19 14:58 bin -> /net/ppm/bin
-rwx------    2 alex     ppm-b         132 Oct  8 14:55 skript.sh

       ||| other
    ||| group
 ||| user
```

Die Datei `skript.sh` ist ein Shell-Skript und ist für den User ausführbar. Der Link `bin` zeigt auf ein anderes Verzeichnis unter `/net/ppm/bin`, `readme.txt` ist eine Textdatei in der nur der User Lesen und Sch´reiben darf und als oberstes steht das Verzeichnis `CVODE`, was an dem `d` an der ersten Stelle zu erkennen ist.

Die Eingabe des Befehls

```
ls i*
```

listet alle Files im aktuellen Pfad auf, deren Namen mit einem i beginnen. Hierbei haben die Sternchen (der sogenannte Joker) die Wirkung, das an der entsprechenden Stelle des Filenamen beliebige Zeichen stehen können.

Probiert den ls-Befehl in verschiedenen Variationen aus. Z.B.:

```
ls
ls -l
ls -l i*
ls -l /homes/itp
ls -l /
```

Der `cp`-Befehl

kommt vom englischen "CoPy" (kopieren). Der cp-Befehl hat im allgemeinen folgendes Format:

```
cp vonFile nachFile
```

Also z.B.

```
cp /etc/motd kopie.motd
```

In diesem Fall würde die Datei `motd` (Message of the day) auf die Datei `kopie.motd` kopiert. Der Befehl

```
cp /etc/motd .
```

kopiert beispielsweise die Datei in den aktuellen Pfad unter demselben FileNamen hier `motd` (der Punkt symbolisiert das aktuelle Verzeichnis).

Der `mv`-Befehl

kommt vom englischen "MoVe" (bewegen). Mit diesem Befehl kann man Dateien umbennen und/oder in ein anderes Verzeichnis verschieben. Der mv-Befehl hat das folgende Format:

```
mv alterdateiname neuername
```

wobei `neuername` entweder der neue Name der betroffenen Datei ist, wenn man die Datie nur umbennen will, oder der Name des neuen Verzeichnisses, in das die betroffene Datei verschoben werden soll. Z.B. durch den Befehl

```
mv meinedatei deinedatei
```

wird die Datei `meinedatei` in `deinedatei` umbenannt. Oder durch

```
mv gedicht kunst/gedichte
```

wird die Datei `gedicht` in das Verzeichnis `kunst/gedichte` verschoben und behält ihren Namen `gedicht`.

Der `pwd`-Befehl

kommt vom englischen "Print Working Directory" (zeige das aktuelle Arbeits-Verzeichnis an). Also z.B. erzeugt

```
pwd
```

die folgende Ausgabe (oder zumindest eine ähnliche)

```
/homes/itp/itp-123/
```

Der `rm`-Befehl

kommt vom englischen "ReMove" (wiederbewegen bzw. löschen).

Erzeugen von Files ist so einfach, daß Ihr bald die Übersicht verliert, wenn Ihr nicht regelmäßig Files löscht, die Ihr nicht mehr braucht. Das vollständige Kommando lautet:

```
rm FileBeschreibung
```

also z.B. `rm *` löscht alle Dateien im aktuellen Verzeichnis.

Es erscheint eine Abfrage in der Form: `rm: remove filename?`

Hier müßt Ihr, wenn Ihr das angezeigte File wirklich löschen wollt, ein " y" eingeben und mit "return" bestätigen. Wenn das File doch nicht gelöscht werden soll, gebt Ihr ein "n" ein (Vorsicht ist kein Standard !!! Um die Sicherheitsabfrage zu erhalten muß eigentlich `rm -i FileName` eingeben werden.). Der letzte Befehl würde jedes File des Directorys abfragen. Wenn Ihr hier immer "y" eingebt, löscht Ihr sämtliche Files des aktuellen Directorys. Deshalb Vorsicht mit dem Sternchen bei dem `rm`-Befehl. Wenn Ihr ein bestimmtes File löschen wollt, benutzt Ihr den `rm`-Befehl am besten in der Form:

```
rm PFUIFILE.c
```

Der `mkdir`-Befehl und der `rmdir`-Befehl

kommt vom englischen "MaKe DIRectory" (mache ein Verzeichnis) bzw. von "ReMove DIRectory" (lösche Verzeichnis).

Mit dem `mkdir`-Befehl erzeugt man ein neues Verzeichnis, z.B.:

```
mkdir aufgaben
```

In diesem Fall wird nun das Verzeichnis `aufgaben` neu erstellt. Diese kann mit dem `rmdir`-Befehl dann auch jederzeit wieder gelöscht werden!

Der `cd`-Befehl

kommt vom englischen ”Change Directory” (wechsele das Verzeichnis).

Mit cd wechselt man das Verzeichnis. Gebt mal folgende Befehle nacheinander ein:

```
pwd
cd /
pwd
cd usr
pwd
cd local
pwd
cd bin
pwd
cd
pwd
```

Natürlich kann man an den jeweiligen Stellen das `pwd` auch weglassen, aber häufig weiß man dann nicht mehr, wo man ist.

Jetzt seid Ihr auf jeden Fall in Eurer home directory. Also, um in die home directory zu wechseln, genügt cd ohne jede weiter Angabe. Um in das übergeordnete Verzeichnis, also eins zurück zu kommen gebt Ihr folgendes ein:

```
pwd
cd ..
pwd
```

Der `man`-Befehl

kommt vom englischen ”MANual” (Anleitung).

Mit Hilfe des `man`-Befehls kann man sich eine ausführliche Beschreibung einzelner UNIX-Kommandos auf dem Bildschirm anzeigen lassen. Gebt doch beispielsweise mal folgendes ein:

`man mkdir`

Mit der Leerzeichen-Taste kann man weiterblättern. Mit `q` (für quit) kann man die `man`–page verlassen.

Mit `man man`

erhältst Du mehr Info's über das Manual-System. Ihr werdet sehen, daß das meiste in englischer Sprache ist. Daran müßt Ihr Euch gewöhnen; Computer und Englisch gehören halt zusammen, wie die Eier zu Ostern.

Der `more`-Befehl

Mit `more FileName` kann man sich den Inhalt des Files FileName anschauen, ohne etwas zu verändern. Auch hier kann man mit der Leerzeichen-Taste weiterblättern und mit `q` abbrechen.

Übung: Schau Dir die `man`-page von `more` an.
Suche in dieser nach dem Ausdruck `pattern` (mehrmals vorhanden !), da kannst Du erfahren, wie man in `man`-pages schnell Schlagwörter findet.

Der `history`-Befehl

Er listet alle Befehle auf, die in dem aktuellen Xterm-Fenster (meist wird die Anzahl der Befehle begrenzt, bei Euch sind es 100) ausgeführt wurden. Wenn Ihr also `history` eingebt erscheint, folgende Anzeige (oder so ähnlich):

```
1 ls
2 cd ..
3 pwd
```

Ihr könnt den jeweils letzten Befehle aus der Historie zurückholen, indem Ihr die Pfeiltaste nach oben drückt. Sobald der gewünschte Befehl erscheint, drückt Ihr einfach RETURN, um ihn erneut auszuführen. Das ist in der Praxis sehr nützlich und Ihr solltet davon gebrauch machen.

4.3 Der Umgang mit Disketten

Um Sicherheitskopieen zu erstellen, oder Daten mit Eurem Computer zu Hause auszutauschen, solltet Ihr selbst mitgebrachte Disketten benutzen. Mit den sogenannten **mtools** könnt Ihr 3 1/2 Zoll Disketten im MS-DOS Format lesen, beschreiben und formatieren. Die wichtigsten Befehle sind im Folgenden aufgelistet:

Befehl	**Beschreibung**
`mdir a:`	Zeigt das Inhaltsverzeichnis auf der Diskette an.
`mcopy -t a:`*filename* `.`	Kopiert das File *filename* von der Diskette in das aktuelle Verzeichnis auf der Festplatte. (*)
`mcopy -t a:'*' .`	Kopiert alle Files von der Diskette in das aktuelle Verzeichnis. (*)
`mcopy` *filename* `a:`	Kopiert das File *filename* aus dem aktuellen Verzeichnis auf die Diskette im Laufwerk.
`mmd a:`*directoryname*	Erzeugt ein MS-DOS Verzeichnis *directoryname* auf der Diskette.
`mrd a:`*directoryname*	Löscht ein MS-DOS Verzeichnis *directoryname* auf der Diskette.
`mcd a:`*directory*	Wechselt in die *directory* auf der Diskette.
`mformat a:`	Formatiert eine Diskette im MS-DOS Format.

(*) Die Option `-t` bei mcopy steht für *translate* und sorgt für die korrekte Umwandlung von MS-DOS Textdateien in das UNIX Format.

5 Shell Skripte

5.1 Einleitendes

Mit den hier erlernten Befehlen und denen aus dem Anhang, lassen sich schon einfache Programme genannt Skripte erstellen. Dazu aber erst später.

Während einer Sitzung am Rechner kommuniziert der Benutzer normalerweise nicht direkt mit dem Betriebssystem, sondern mit einem Programm, welches seine Kommandos liest, analysiert und dann entweder selbst ausführt oder an andere Programme weiterreicht. Dieses Programm wird deshalb als Kommandointerpreter bezeichnet und trägt den Namen Shell, weil sie wie eine Schale um den Kern des Betriebssystems liegt.

Auf eurem Desktop versteckt sich die Shell in dem Xtem Fenster. Wenn ihr dort also Befehle eingebt, sendet Ihr also eigentlich Befehle an die Shell. Auf den unterschiedlichen UNIX-Implementierungen existiert eine Reihe verschiedener Shell-Programme. Die Bourne-Shell, benannt nach ihrem Autor, ist auf praktisch allen Systemen vorhanden (in dieser Einleitung wird fast nur diese Shell behandelt). Eine zweite, sehr weit verbreitete Shell, die C-Shell, entwickelt an der Universität von Californien in Berkeley, stellt eine beträchtliche Erweiterung der Bourne-Shell dar, ist aber nicht auf allen UNIX-Systemen vorhanden und auch noch nicht so weit normiert.

Man kann während einer Sitzung jederzeit eine Subshell aufrufen, indem man das Kommando

```
sh
```

für eine Bourne-Shell oder das Kommando

```
csh
```

für eine C-Shell eingibt. Mit dem Kommando

```
exit
```

wird diese Shell wieder verlassen und man befindet sich in der aufrufenden Shell. Gibt man das Kommando exit in der login-Shell, so ist dies gleichbedeutend mit einem Logout. In einem gesicherten Moment kann man das durchaus einmal ausprobieren

Es gibt noch mehr Arten von Shells, die aber immer auf eine der beiden Grundshells zurückgehen.

5.2 Das Skript

Wie sieht nun der Aufbau eines Skripts aus?

Skripte sind einfach Textdateien, welche UNIX-Befehle enthalten. Es darf immer nur ein Befehl pro Zeile stehen. Die Befehle werden von oben nach unten vom Shell-Interpreter abgearbeitet. In der ersten Zeile steht die aufzurufenden Shell.

```
#!/bin/sh
```

Für die Bounce-Shell, die man auch möglichst immer verwenden sollte, da sie auf fast allen UNIX-Linux Systemen installiert sein wird. Ein einfaches Skript könnte demnach lauten:

```
#!/bin/sh

cd /homes/ppm
ls -l
```


Was macht dieses Skript? In der ersten Zeile wird eine Bourne-Shell geöffnet. In dieser neuen Shell läuft der Shell-Interpreter also von oben nach unten und trifft auf die zweite Zeile. Mit `cd /homes/ppm` wechselt man in das entsprechende Verzeichnis und `ls -l` gibt eine Übersicht üeber das entsprechende Verzeichnis aus.

Um ausführbare Dateien und nicht ausführbare Dateien zu unterscheiden, gibt es unter UNIX das Dateirechtesystem (siehe Kapitel 4). Dieses ordnet jeder Datei eine ganze Reihe von Rechten zu. Wenn man eine Datei mit dem Editor erzeugt hat, ist die Einstellung so, daß sie als eine nicht ausführbare Datei gekennzeichnet ist. Um diesen Zustand zu ändern und ein Skript mit Leben zu füllen, brauchen wir einen neuen Befehl `chmod u+x showppm`. Dieser setzt für den User (also Ihr) die Rechte der Datei `skript` auf ausführbar. Mit der Eingabe von `showppm` bekommt man dann die Ausgabe des Skriptes.

Shell-Skripte können ein mächtiges Werkzeug sein. Die Shell (`sh` und `csh`) stellt eine komplette Programmiersprache dar. Es gibt die Möglichkeit Variablen zu deklarieren, Kontrollstrukturen und Schleifen anzuwenden und vieles mehr. Allerdings ist der Aufwand zum erlernen vergleichbar mit dem eine neue Programmiersprache zu erlernen, deshalb wird hier zugunsten anderer Schwerpunkte auf kompliziertere res verzichtet.

Übung: Shell Skript.
Erstellt ein Skript, welches egal, wo man sich befindet die Lage des eigenen Home-Verzeichnisses mit `pwd` und eine Auslistung aller Dateien in eben diesem Verzeichnis mit `ls` ausgibt.

Hinweis: Wer jetzt nicht weiß was zu tun ist, ersteinmal das Skript weiter durcharbeiten und dann später an diese Stelle zurückkehren.

6 Drucken

Früher glaubte man, daß mit der Einführung des Computers und der Möglichkeit Informationen digital abzuspeichern, der Papierverbrauch drastisch zurückgehen werde und die Rettung des Waldes schon in greifbarer Nähe sei. Leider hat sich diese Hoffnung nicht erfüllen können

Ob es sinnvoll ist ein Programm oder Skript auszudrucken müßt Ihr im Einzelfall selbst entscheiden, es sei da an Eure Vernunft appelliert. Allerdings besteht eine Obergrenze von 100 Blatt im **Semester**

Die Befehle zum Drucken auf den einzelnen Druckern lauten wie folgt:

Drucker	**Befehl**		**Filetypen**
Laserdrucker	`lpr -Ppss`	*filename.ps* *filename*	Postscript Files Text Files, Listings ... einseitig bedruckt
	`lpr -Ppsd`	*filename.ps* *filename*	Postscript Files Text Files, Listings ... beidseitig bedruckt
Plotter	`lpr -Ppl1`	*filename.hpgl*	HPGL-Files
DVI-Drucker	`lpr -Pdvi`	*filename.dvi*	mit LaTeX erstellte DVI-Files

Übung: Verändere Dein File und lasse es ausdrucken

Du kennst jetzt die wichtigsten Befehle, um Text auf Files zu schreiben, um Files in den Editor zu lesen, sie anzuschauen oder zu verändern. Wichtig ist noch das Ausdrucken. Dazu geht ihr mit der Maus wieder in ein Xterm-Fenster und gebt ein:

`lpr` *EuerFileName*

Mit dem `lpq`-Befehl könnt Ihr dann kontrollieren, ob Eure Druckanweisung auch beim Drucker angekommen ist. Wenn Ihr den Matrixdrucker kontrollieren wollt, gebt einfach

`lpq`

ein. Für den Plotter und den DVI-Drucker müßt Ihr zusätzlich die jeweilige Druckerspezifikation angeben. Also

`lpq -Ppl1` für den Plotter bzw.
`lpq -Pdvi` für den DVI-Drucker.

Es gibt momentan zwei postskriptfähige Laserdrucker, in zwei verschiedenen Räumen. Mit dem Befehl `lpr -Ppsd` kommen die Druckaufträge immer auf dem Drucker an, der sich in dem selben Raum befindet wie Ihr Euch selbst.

7 Der Editor nedit

Der Editor-nedit ist ein Schreibprogramm, mit dem man Dateien schreiben, lesen und verändern kann. Er ersetzt sozusagen die Schreibmaschine. Um diesen Editor besser kennenlernen könnt, haben wir eine Datei vorbereitet, die Ihr zunächst in Euren Bereich kopieren müßt. Dieses macht Ihr folgendermaßen:

Gebt auf UNIX-Ebene, d.h. in dem weißen x-term Fenster, folgendes ein und beendet den Befehl dann mit RETURN:

`cp ~ppm-soft/pub/intro/nintro .`

Auf diese Weise habt Ihr Euch nun die Datei „nintro“ in Euer aktuelles Verzeichnis kopiert. (`cp` steht für *copy* und der Punkt für das aktuelle Verzeichnis))
Diese Datei sollt Ihr Euch nun mit Hilfe von nedit ansehen, indem Ihr diesen folgendermaßen aufruft und ihm gleichzeitig mitteilt, welche Datei Ihr bearbeiten wollt:

`nedit nintro` &

Auch diesen Befehl müßt Ihr natürlich wieder mit einem RETURN abschließen.

Es ist wichtig, daß Ihr auch das sogenannte „Kaufmannsund“, &, am Ende dieses Befehls eingebt, denn es bewirkt, daß der Befehl im Hintergrund ausgeführt wird. Auf diese Weise vermeidet Ihr, daß das x-term Fenster blockiert wird, d.h. Ihr könnt ohne Probleme in dem x-term Fenster weitere Befehle eingeben.
So, nun viel Spaß beim Durcharbeiten des „nintro“ und Kennenlernen von nedit! Druckt Euch dieses File bitte nicht aus, da dieses nur einen unnötigen Papierberg erzeugen würde. Nur wenn Ihr Euch sicher seid, daß Ihr dieses Intro unbedingt braucht, um mit nedit zurechtzukommen, druckt Euch maximal ein Exemplar je Gruppe aus!

Im folgenden werden wir erklären, wie Ihr ein Programm übersetzt und zum Laufen bringt.

8 Mein erstes Programm

Um ein erstes Programm zu bekommen kopiert Ihr Euch ein bereits fertiges Programm in Euren Bereich. Das macht Ihr mit folgenden Befehlen, je nachdem, ob Ihr Pascal, C oder FORTRAN 95 lernen wollt:

Sprache	Eingabe
C	`cp ~ppm-soft/pub/intro/beispiel.c .`
FORTRAN	`cp ~ppm-soft/pub/intro/beispiel.f .`
Pascal	`cp ~ppm-soft/pub/intro/beispiel.p .`

Mit `ls` schaut Ihr nach, ob das File in Eurem Verzeichnis angekommen ist. Schaut Euch den Inhalt des Files an, ohne ihn zu verändern (also mit `more FileName`).

Um das Programm ausführen zu lassen, müssen wir es noch in eine dem Computer verständliche Sprache übersetzen - auch **compilieren** genannt.

`gcc, xlf95` oder `gpc` sind die Namen der jeweiligen Compiler, dem ein Programmname und evtl. eine Compileroption folgen(`-lm` steht für das Einbinden einer mathematischen Bibliothek(library)).

```
gcc beispiel.c -lm
xlf95 beispiel.f
gpc beispiel.p.
```

Der Compiler erstellt ein für den Computer ausführbares File mit dem Namen `a.out`.

Ihr könnt das überprüfen, indem Ihr auf UNIX-Ebene mit dem Befehl `ls` die aktuelle Fileliste in Eurem Bereich ausgeben laßt. Das sollte ungefähr so aussehen:

```
itp-123 <6>% ls
a.out    beispiel.c
```

Indem Ihr `-o ProgramName` in dem Feld Optionen eintragt und dann neu compiliert, könnt Ihr dem Compiler sagen, wie das ausführbare Programm anstatt `a.out` heißen soll.

Die Befehlszeile sieht dann folgendermaßen aus:

```
gcc -o ProgrammName beispiel.c -lm
xlf95 -o ProgrammName beispiel.f
gpc -o ProgrammName beispiel.p
```

Es wird ein ausführbares Programm mit dem Namen `ProgrammName` erzeugt. Das Programm `beispiel` kann jetzt auch einfach ausgeführt werde, indem man auf der UNIX-Ebene den Befehl: `ProgrammName` mit anschließendem RETURN eingibt.

9 In 8 Sekunden um die Welt: Electronic-Mail

Mit dem Bereich in unserem UNIX-System habt ihr auch automatisch eine
E-Mail-Adresse. Das sind diese Kürzel mit dem komischen @–Zeichen in der Mitte, die wichtige Leute auf ihrer Visitenkarte haben. Diese Zeichen nennt man Klammeraffe (engl. at) und es trennt den Benutzernamen von einer Rechneradresse.

Eure E-Mail-Adresse für die Zeit, die Ihr bei uns angemeldet seid, lautet:
xxx@pool.math.TU-Berlin.DE.
Dabei müßt Ihr für das xxx Euren Login-Namen einsetzen, also z.B. `itp-123`, oder `ppm-201`.
Allgemein ist so eine Adresse immer aus dem Benutzernamen und der Rechneradresse zusammengesetzt. An dem letzten Teil der Rechneradresse könnt Ihr erkennen, aus welchem Land die Adresse stammt. Dabei steht **DE** für Deutschland, andere Beispiele sind **NZ** für Neuseeland oder **PY** für Paraguay.

Wenn der kleine Briefkasten in der rechten oberen Ecke Eures Bildschirms schwarz gefärbt ist, ist für Euch eine neue Mail angekommen.

Um Mail zu lesen oder zu verschicken, gebt Ihr folgenden Befehl ein:

```
pine
```

Euer Bildschirm könnte ungefähr wie folgt aussehen:

Für jede Mail wird jeweils eine Zeile angezeigt, z.B.:

```
Status  Nummer   Datum     Absender                      Betreff
  N        2     Apr  2    TeilnehmerIn itp    (342)    Party
```

Wobei der Status N für Neu steht.

In den unteren beiden Zeilen seht Ihr das Menü, aus dem Ihr die speziellen Pine-Befehle ablesen könnt. Da in diesen beiden Zeilen nicht genug Platz für alle möglichen Befehle ist, gibt es den Befehl **O** für Other commands.
Weitere wichtige Befehle sind die folgenden:

V um sich eine E-Mail anzuschauen bzw. zu lesen (View)

C um selbst eine E-mail zu schreiben (Compose)

R um eine E-Mail zu beantworten (Reply)

I um in die Übersicht zurückzukommen (Index).

Zusätzlich könnt Ihr auch noch RETURN und die Pfeiltasten ↑ und ↓ verwenden. Mit Hilfe der Pfeiltasten könnt Ihr Euch in der Liste Eurer E-Mails hoch- und runterbewegen. Wenn Ihr so die Zeile einer speziellen Mail schwarz markiert habt, dann könnt Ihr diese Mail lesen, indem Ihr einfach RETURN drückt.

Übung: Wir schicken unserer Nachbargruppe eine Mail

Dafür müssen wir uns erst mal den Benutzernamen der entsprechenden Gruppe erfragen.

1. Wir rufen das Mailprogramm **pine** auf.
3. **C** für Compose new Mail drücken.
4. Im Feld **To:** die E-Mail-Adresse der Nachbargruppe eingeben, z.B. itp-130@pool.math.TU-Berlin.DE (das ist nur eine Beispieladresse, jetzt bitte nicht alle an die arme Gruppe itp-130 mailen.)
5. Mit den Pfeiltasten auf das Feld **Subject:** gehen und den Betreff der Mail eingeben.
6. Mit den Pfeiltasten runter in den Eingabebereich gehen und einen beliebigen Text eintippen.
7. Um die Mail abzuschicken, drückt und haltet Ihr erst die CTRL–Taste und drückt dann gleichzeitig X. (Anmerkung: Das Dach-Zeichen unten in der Menüleiste steht jeweils für die CTRL-Taste.)
8. Auf die Frage **Send message?** gebt Ihr **Y** für Yes ein.

Nach kurzer Zeit sollte die Mail bei der Nachbargruppe eintreffen und der Briefkasten schwarz werden. Wenn Ihr selbst gerade eine neue Mail bekommt und Ihr befindet Euch bereits in `pine`, dann dauert es eine Weile bis er das mitbekommt und Euch die Mail anzeigt. Ihr könnt ihn aber explizit dazu auffordern neue Mails anzuzeigen, wenn Ihr CTRL–L dückt.

Genauso wie an Eure Nachbargruppe könnt Ihr an jeden anderen in der Welt E-Mails verschicken oder Euch schicken lassen, Ihr müßt nur seine Adresse kennen.

Übung Schickt eurem Tutor eine E-Mail!

Aufgabenstellung:

Nachdem Ihr alle fleissig dieses Intro durchgearbeitet habt, ist es nun an der Zeit, sich in die Programmiersprachen C bzw. Pascal einzuarbeiten. Deswegen schickt jetzt eine E-Mail an einen Eurer Tutoren, damit dieser sieht, daß Ihr das Ende des Intros erreicht habt und jetzt das C- bzw. Pascalskript benötigt. Dann sprecht ihn an, und er wird, nachdem er Eure E-Mail gelesen hat, Euch Eure Programmiersprachskripte geben.

Diese Aufgabe ist u.a. auch wichtig, damit wir Tutoren wissen, daß Ihr alle mit dem Mailprogramm Pine zurechtkommt, da wir alle aktuellen Mitteilungen immer via E-Mail an Euch schicken werden!!!

Da Ihr für diese Aufgabe die E-Mail-Adresse Eurer Tutoren wissen müßt, verraten wir Euch, wie man die E-Mail-Adresse einer Person herausfindet, wenn man den Namen der betreffenden Person kennt.

Der **finger**-Befehl:

Zum Beispiel findet man mit der folgenden Befehlszeile u.a. die E-Mail-Adresse von Donald Duck heraus:

```
finger Donald Duck
```


Dann erscheint u.a. auch folgendes auf dem Bildschirm:
`Login name: donald.duck`
Also `donald.duck@pool.math.tu-berlin.de` ist dann die E-Mail-Adresse von Donald.

10 WWW - das World Wide Web

Der Vollständigkeit halber und nicht ganz uneigennützig wollen wir noch auf das **World Wide Web** hinweisen.
Es ist zur Abwechslung mal eine europäische Entwicklung und wurde am Kernforschungszentrum CERN in Genf entwickelt.
Das **WWW** kann man sich als ein weltweites Netz bestehend aus miteinander verknüpften Dokumenten vorstellen. Um sich in diesem Netz zu bewegen, braucht man einen sogenannten **WWW-Browser**.
In unserem System steht dafür:

```
netscape
```

zur Verfügung.

WWW-Adressen lauten z.B. `http://www....` und stehen heutzutage unter jeder besseren Werbung[3].
Aber auch die neuesten Nachrichten findet man im **Web:** `http://www.tagesschau.de` oder `http://www.tagesthemen.de` !
Die `homepage` der PPM findet Ihr schließlich unter

`http://www.math.tu-berlin.de/ppm`

Die Bedienung erfolgt fast ausschließlich über die Maus.
Alle hervorgehobenen Wörter oder Bilder in einem Dokument sind ein Verweis (Link) auf ein anderes Dokument irgendwo im Netz. Da das WWW auf dem Internet basiert und dieses notorisch überlastet ist (von wegen Datenautobahn), kann es, je nachdem wo Ihr Euch gerade befindet, mitunter sehr lange dauern.

Wenn Ihr einen der Browser startet, bekommt Ihr die Homepage des UNIX-Pools angezeigt. Darin findet Ihr weitergehende Informationen, die diese Einführung nicht abdeckt und ein paar interressante Links zu anderen WWW-Seiten.

Aber probiert es am besten selbst mal aus! Also ruft einfach Netscape auf, indem Ihr den Befehl `netscape` auf der UNIX-Ebene eingebt!

Um die Suche im Internet sinnvoll zu gestalten, gibt es verschiedene Möglichkeiten. Man kann mit Suchmaschinen oder thematischen Verzeichnissen arbeiten.

Ein Beispiel für eine Suchmaschine ist:

AltaVista `(www.altavista.com)`

Ein Beispiel für ein thematisches Verzeichnis ist:

Yahoo! `(www.yahoo.de)`

Der Unterschied liegt in der Arbeitsweise.
Bei Suchmaschinen durchkämmen Programme (sog. Robots oder Spiders) das Web. Diese laden sehr viele Websites und speichern diese in Datenbanken ab.
Thematische Verzeichnisse hingegen werden von einer Redaktion gepflegt, die einen Katalog über möglichst alle Themen (wie z.B. Gelbe Seiten) erstellt.

[3] Das Maggi-Kochstudio hat auch eine: http://www.maggi.de/maggi/

Suchstrategien:

AltaVista: Die Anfrage sollte möglichst speziell ausfallen, da sonst die Trefferausgabe sehr groß sein kann. Um Wörter zu verknüpfen oder andere Einschränkungen vorzunehmen, können in der **Advanced Search** folgende Optionen eingegeben werden:

Jokerzeichen:	*
logische Operatoren:	AND,OR,NOT,NEAR
Phrasensuche:	"..."
Dokumententitel:	title:...
Servername:	host:...
Serverdomain:	host:de,nl,com...
URL:	url:...
Link:	link:www....
Dateityp:	link:mp3,wav...

Yahoo!: Es gibt zwei Möglichkeiten entweder blättert man von der Startseite aus in verschiedene Kategorien oder man sucht direkt nach einem Schlagwort.

Übung: Findet im Web die 'Kleine Einführung in Unix'.
Tip: Verwendet 'Phrasensuche' und 'Jokerzeichen'.

11 Anhang

Hilfreiche Unix-Befehle (für den Pool im MA 241)

Befehle	Erklärungen
`a2ps` *FileName* > *File.ps*	Erzeugen einer Postscript-Datei aus einer ASCII-Datei
`cat` *File1 File2* > *File3*	Hängt File2 an File1 und schreibt alles in File3
`cd` *DirName*	Wechseln des Arbeitsverzeichnisses
`cp` *vonPfad/File nachPfad/File*	Kopieren einer Datei
`dvips` *File.dvi* > *File.ps*	Erzeugen einer Postscript-Datei aus einer dvi-Datei
`find . -name` *FileName1* > *FileName2*	Suchen nach der Datei FileName1 in allen Unterverzeichnissen und Trefferausgabe in FileName2
`finger` *UserName*	Ausgabe von Informationen über den Benutzer UserName
`grep [-option]` *zeichenkette FileName*	Durchsuchen von Dateien auf bestimmte Zeichenketten
`head [-n]` *FileName*	Ausgabe der ersten n-Zeilen von FileName
`kill [`*PID*`]`	Abbrechen des Prozesses mit der Prozeß-ID
`less` *FileName*	Auflisten der Datei FileName
`lpq [-P`*printer*`]`	Anzeige der Druckaufträge
`lpr [-P`*printer*`]` *FileName*	Ausdrucken von FileName
`lprm [-P`*printer*`]` *JobNummer*	Löschen des Druckauftrages
`ls [-option]`	Auflisten aller Dateien und Verzeichnisse im aktuellen Verzeichnis
`man` *Kommando*	Ausführliche Beschreibung eines Kommandos
`mkdir` *DirName*	Erstellen vom Verzeichnis DirName
`more` *FileName*	Auflisten der Datei FileName
`mv v`*onPfad/File nachPfad/File*	Verschieben einer Datei
`ps [-option]`	Auflisten der momentan laufenden Prozesse
`pwd`	Anzeigen des aktuellen Pfades
`rm -i` *FileName*	Löschen von FileName
`rmdir` *DirName*	Löschen des Verzeichnisses DirName

`rwho -a`	Liefert eine Liste der im Unix-Pool eingeloggten Benutzer
`sort [-option]` *FileName*	Sortieren der Zeilen einer Datei nach unterschiedlichen Kriterien
`ssh` *RechnerName*	Einloggen auf RechnerName
`tail [-n]` *FileName*	Ausgabe der letzten n-Zeilen von FileName
`wc` *FileName*	Liefert die Anzahl der Zeilen, Wörter und Zeichen von FileName
`xgal`	Ein kleines Spiel

Hilfreiche Shortcuts für die Shell

Crtl + c	Abrechen eines Programmes
Crtl + d	Schließen einer weißen Shell, in der schwarzen Shell = `logout`
Crtl + l	Reset einer Zeile und Terminal refresh
Crtl + w	Löschen einer Zeile
Crtl + z	Anhalten eines Programmes. Mit `bg` kann man das Programm dann im Hintergrund laufen lassen. Nützlich wenn man & vergessen hat !

Hilfreich für die Anwendung der Befehle ist das Lesen der `man`-page oder Nachlesen im Netz, z.B. unter:

`http://irb.cs.tu-berlin.de/leitfaden/sw-dokumente/unix/`