Zeitschrift: Visionen : Magazin des Vereins der Informatik Studierenden an der ETH Zürich

Herausgeber: Verein der Informatik Studierenden an der ETH Zürich

Band: - (1994)

Heft: 6

Heft

Nutzungsbedingungen

Die ETH-Bibliothek ist die Anbieterin der digitalisierten Zeitschriften auf E-Periodica. Sie besitzt keine Urheberrechte an den Zeitschriften und ist nicht verantwortlich für deren Inhalte. Die Rechte liegen in der Regel bei den Herausgebern beziehungsweise den externen Rechteinhabern. Das Veröffentlichen von Bildern in Print- und Online-Publikationen sowie auf Social Media-Kanälen oder Webseiten ist nur mit vorheriger Genehmigung der Rechteinhaber erlaubt. Mehr erfahren

Conditions d'utilisation

L'ETH Library est le fournisseur des revues numérisées. Elle ne détient aucun droit d'auteur sur les revues et n'est pas responsable de leur contenu. En règle générale, les droits sont détenus par les éditeurs ou les détenteurs de droits externes. La reproduction d'images dans des publications imprimées ou en ligne ainsi que sur des canaux de médias sociaux ou des sites web n'est autorisée qu'avec l'accord préalable des détenteurs des droits. En savoir plus

Terms of use

The ETH Library is the provider of the digitised journals. It does not own any copyrights to the journals and is not responsible for their content. The rights usually lie with the publishers or the external rights holders. Publishing images in print and online publications, as well as on social media channels or websites, is only permitted with the prior consent of the rights holders. Find out more

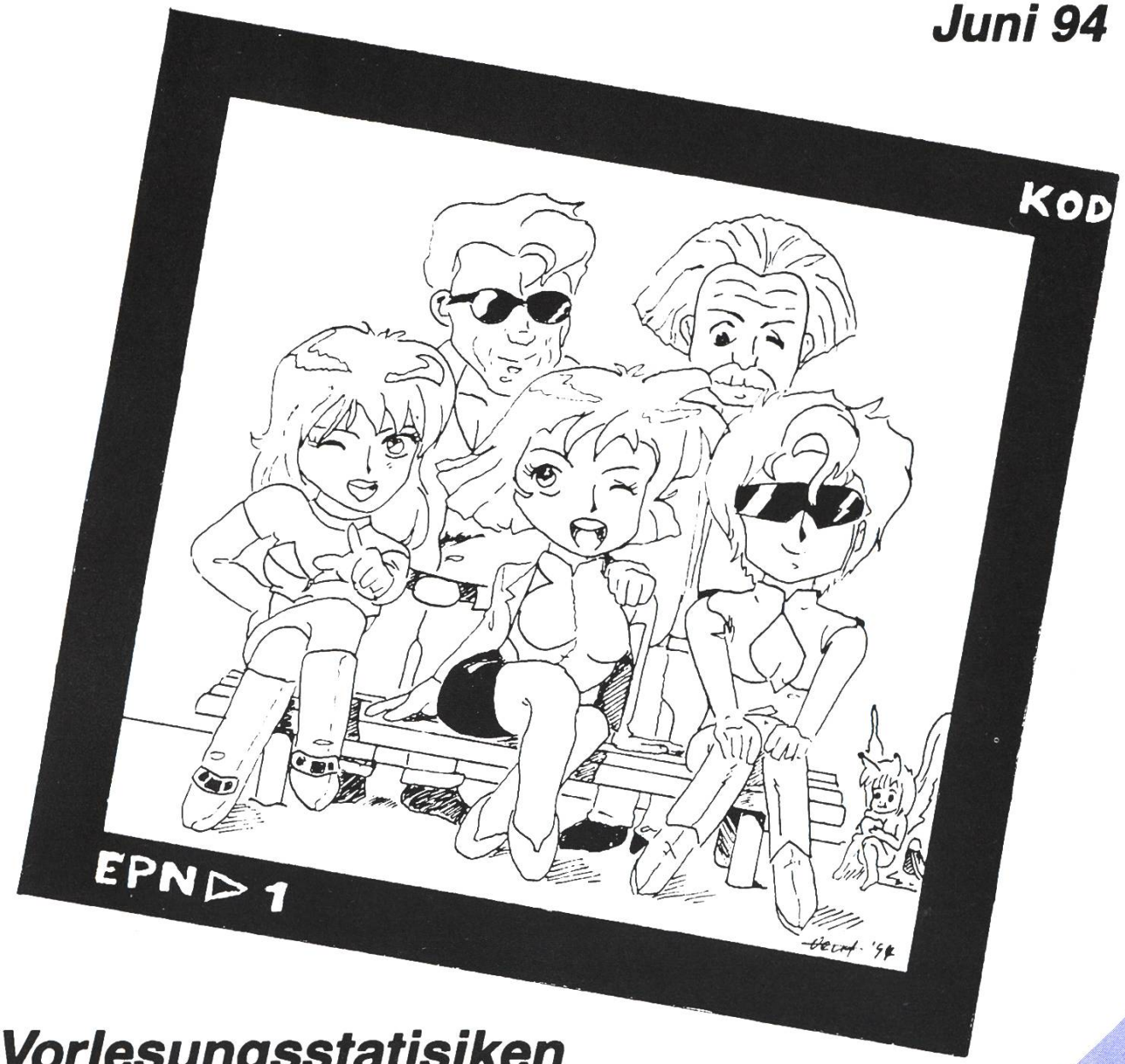
Download PDF: 20.08.2025

ETH-Bibliothek Zürich, E-Periodica, https://www.e-periodica.ch

Visionen

6

Juni 94



Vorlesungsstatisiken
C-Kurs, vierter Teil
FTP, Archie, WWW

Visionen Herausgegeben vom Verein der Informatikstudierenden an der ETH Zürich

VISKAS

Adressen

Aktuar: Stefan Rohmer
Keltenstrasse 6, 8044 Zürich
Tel. 01 / 251 34 51
e–mail: stefan@vis.inf.ethz.ch

Exkursionen: Frank Möhle
Dielsdorferstrasse 7, 8155 Niederhasli
Tel. 01 / 851 03 21
e–mail: fmoehle@iiic.ethz.ch

Feste & Kultur: Christian Limpach
Mainaustrasse 44, 8008 Zürich
Tel. 01 / 422 27 13
e–mail: chris@vis.inf.ethz.ch

Präsidentin: Grete Danielsen
Dohlenweg 26, 8050 Zürich
Tel 01 / 302 48 97
e–mail: gcdaniel@iiic.ethz.ch

Prüfungen und Unterricht:
Leonhard Jaschke
Südstrasse 67, 8008 Zürich
Tel. 01 / 383 60 55
e–mail: ljaschke@iiic.ethz.ch

Quästor: Daniel Kluge
Irringersteig 3, 8006 Zürich
Tel. 01 / 252 04 14
e–mail: dankluge@iiic.ethz.ch

Redaktorin: Nathalie Weiler
Säntisstrasse 5, 8008 Zürich
Tel. 01 / 381 63 01
e–mail: nathalie@vis.inf.ethz.ch

Verleger: Hans Domjan
Kapfhalde 3, 6020 Emmenbrücke
Tel. 041 / 53 68 83
e–mail: hdomjan@vis.inf.ethz.ch

Visinfo(Infosystem): Michel Müller
Rheinländerstr. 15, 4056 Basel
Tel. 061 / 321 81 23
e–mail: mimuelle@iiic.ethz.ch

Impressum

Herausgeber:
Verein der Informatikstudierenden an der ETH Zürich.

Verleger: Hans Domjan
Redaktorin: Nathalie Weiler

Adresse Verlag & Redaktion:
VIS
Verein der Informatikstudierenden
IFW B29
ETH Zentrum
8092 Zürich

Tel: 01 632 72 12 (Mo–Fr, 12.15–13.00)
Fax: 01 262 39 73
e–mail: vis@iiic.ethz.ch
Postkonto 80–32779–3
Präsenzzeit: Mo–Fr: 12.15–13.00

Auflage: 1300
Inseratepreise:

1 Seite	s/w	SFr. 500.–
1 Seite	Farbe	SFr. 750.–
1/2 Seite	s/w	SFr. 250.–
1/4 Seite	s/w	SFr. 150.–

Redaktions– und Anzeigeschluss für die nächste Ausgabe:
Freitag, 17. Juni 1994

Visionen

© 1994 by
Verein der Informatikstudierenden

Hei Folkens!

In der letzten Zeit ist im Vorstand wieder einiges gelaufen (trotz meiner zweiwöchigen Ferien im Hohen Norden!) Lasst mich das Wichtigste kurz zusammenfassen:

Die Vorbereitungen für unserern ACM-Wettbewerb laufen auf Hochtouren. Das OK setzt sich zusammen aus Frank (Contest Director), Christian (Chief Judge) und Hans (Contest Secretary). Wie mir gesagt wurde, findet die ETH-interne Ausscheidung am 5.11.94, der South West European Contest am 11./12.11.94 (an der ETH !!) statt. Für die ETH-interne Ausscheidung werdet Ihr in den nächsten Visionen Infos und Hinweise zur Anmeldung finden. In diesem Zusammenhang hat der VIS die vor sechs (!!) Jahren begonnenen Bemühungen, Student Chapter of the ACM zu werden, wieder aufgenommen, und die Chancen dafür stehen sehr gut.

Wer nicht bis November auf das grosse Event warten will, dem (und natürlich auch allen anderen :-) empfehlen wir die Giga-Fete dieses Semesters, nämlich das VISKAS. Es findet am 16. Juni (Donnerstag) statt. Wie immer am Katzensee (Wir hoffen, dass das Wetter hält). Eingeladen sind auch alle Professoren und Assistenten. Es hat genügend Spanferkel.

Wenn wir schon bei den Profs sind, möchten wir vom VIS an dieser Stelle Herrn Prof. J. Gutknecht für seine Wiederwahl als Abteilungsvorsteher IIIC herzlich gratulieren und wir hoffen auf die Fortsetzung der angenehmen Zusammenarbeit.

Für diejenigen unter Euch, die sich schon um eine künftige Anstellung kümmern, haben wir zwei Veranstaltungen anzubieten: Zum einem findet "Bewerben - aber richtig" (wurde schon ein paar mal angekündigt und dann ein paar mal verschoben...) nun am 20.Juni 1994 definitiv statt. Anmeldung ist keine erforderlich, der Vortrag beginnt um 17:15 im AudiMax. Andererseits wird das von SIA und GII organisierte Kontaktgespräch "Diplom - was nun?" zwischen Studierenden (IIIA bis E) und Wirtschaftsvertretern am 14.Juni ab 18:30 durchgeführt. Dazu müsst Ihr Euch allerdings beim VIS anmelden und ein Depot von CHF 20,- hinterlegen, das Ihr am selbigen Abend wieder zurück bekommt. Die Teilnehmerzahl ist auf 10 beschränkt. First Come, First Served, wie immer.

Diesen Frühling haben wieder mal Prüfungen stattgefunden. Die kommentierten Statistiken im Detail findet Ihr weiter hinten. Leider hat es bei diesem Event unseren Exkursionisten Boris erwischt. Wir wünschen ihm in der Zukunft viel Glück und Erfolg. (Nå har jeg dessverre ingen jeg kan snakke norsk med på styremøtene ...). Somit rutscht Christain Limpach (4. Sem.) nach und übernimmt Frank's Ressort "Feste und Kultur". Frank seinerseits wird in Zukunft das Ressort Exkursionen betreuen.

Damit Ihr in Zukunft jederzeit elektronisch Zugriff auf den VIS habt, wurde ein experimenteller WWW-Server (World Wide Web) eingerichtet. Er kann durch folgende Eingabe in Mosaic erreicht werden: `http://vis-next/` . Weiteres zu WWW in diesen Visionen.

Jeg håper dere alle får en fin vår før stresset starter til sommerern igjen.

Grete

ETH Zürich IIIE
Einladung

Die interessierten Studierenden im 4. Semester der Abteilung IIIA, IIIB, IIIC, IIID, sind eingeladen zur

Orientierung über das Diplomstudium (5. - 8. Semester) in

Betriebs- und Produktionswissenschaften

an der Abteilung IIIE.

Prof. J. Reissner - der Abteilungsvorsteher IIIE - informiert über das Studium und beantwortet Fragen der Studierenden, die an einem Wechsel in die Abteilung IIIE interessiert sind.

Dominik Bär - Studierender des vierten Studienjahrganges - berichtet über seine Erfahrung an der Abteilung IIIE.

Datum	Mittwoch, 8. Juni 1994
Zeit	13.15 - 14.45 Uhr
Ort	ML H 44

Präsenzen im VIS-Büro: von 12:15 bis 13:00

Montag : Nathalie
Dienstag : Hans
Mittwoch : Christian
Donnerstag : Daniel
Freitag : Grete oder Stefan

Feste und Kultur: Wechsel im (wichtigsten) Ministeramt

Wie auch in der Politik wechseln auch im VIS die Minister von Zeit zu Zeit. Doch nicht wegen irgendwelchen Bestechungsskandalen oder ähnlichen Zwischenfällen, sondern einfach so. Deshalb kann ich auch an dieser Stelle meinen Rücktritt selber und mit gutem Gewissen bekanntgeben.

Doch wer nun denkt, der VIS wäre mich nun für immer los, der hat sich gewaltig geirrt; ich bleibe. Nur werde ich mich in Zukunft nicht mehr um die legendären VIS–Feste kümmern, sondern werde die Exkursionen in die Industrie und Forschung in die Wege leiten.

Wenn Ihr mich nun des öfteren auf Exkursionen seht ist das kein Zufall mehr. Genaue Pläne habe ich noch nicht, aber die erfahrt Ihr dann bestimmt in den nächsten Visionen.

Aber auch in Zukunft wird man mich immer noch an den Festen sehen. Das Amt hat mir zu viel Spass gemacht, als dass ich es einfach von Heute auf Morgen einfach so fallen lassen könnte. Und damit die Qualität der Feste mindestens gleich bleibt, dafür sorgt der neue Festminister, der das Festdiplom am letzten Viskas erarbeitet hat. Christian Limpach war aktiv im OK tätig und für die Finanzen verantwortlich.

Ich hoffe, dass all die treuen Helferinnen und Helfer, die mir immer beiseite standen und mich in den verschiedensten Situationen vor dem Ueberarbeiten gerettet haben und mich mit vielen guten Ideen versorgt haben, auch Christian in gleicher Weise unterstützen werden.

In diesem Sinne wünsche ich Dir, Christian alles Gute und viel Erfolg in Deinem neuen Amt; Spass wirst Du auf jeden Fall daran haben!

fm

Aus der Vorlesung

"Ein Würfel mit 7 Seiten hat eine grössere Unsicherheit als ein Würfel mit 6 Seiten"
– Prof. Maurer in TI II 28.4.98

"Ich muss sie warnen, die meisten Programme, die ich schreibe sind falsch, aber das geht nicht nur mir so"
– Prof. Engeler Rekursivität und Komplexität 28.4.94

"Kreditoren sind Leute, die glauben, sie bekaemen noch etwas vonuns..."
– Prof. Senti Einführung in Betriebswirtschaft 25.4.94

Das Internet und seine Dienste

Da das Internet seit diesem Semester auch den Informatik–Studenten ab dem dritten Semester weltweit offen steht, lohnt es sich, die wichtigsten Dienste auszuprobieren. Ich möchte hier grob beschreiben, was dabei abläuft, damit diese Dienste besser verstanden und verantwortungsvoll genutzt werden.

Anonymes FTP

Ftp (File Transfer Protocol) ist ein altes Protokoll, um Dateien zwischen zwei Rechnern zu kopieren, die miteinander verbunden sind. Dabei meldet man sich mit einem Benutzernamen und einem Passwort beim entfernten System an und kann dann mit einfachen Befehlen das Inhaltsverzeichnis ansehen und Dateien in beide Richtungen kopieren.
Besonders interessant ist der anonyme Betrieb auf dem Internet. Denn an vielen Orten auf der ganzen Welt werden grosse Datenbestände öffentlich zugänglich gemacht. Das können Shareware– und Public–Domain–Programme, kleinere Updates für kommerzielle Software, Satellitenbilder, Forschungsprototypen und vieles mehr sein. Der Benutzername ist dabei immer ftp (oder anonymous); das Passwort ist die eigene Email–Adresse. Eine Sitzung kann so aussehen:

```
rif8% ftp nic.switch.ch
220- Hello user at
rif8.iiic.ethz.ch [129.132.13.18]
Name (nic.switch.ch:mbleiche):
ftp
331 Guest login ok, give your
e-mail address (user@domain) as
password.
Password: mbleiche@iiic.ethz.ch
230- Guest
'mbleiche@iiic.ethz.ch' login ok.
ftp> cd /mirror/info-mac/comm
ftp> ls
00comm-abstracts.abs
...
zz-woof-1b19.hqx
ftp> bin
ftp> get zterm-09.hqx
ftp> quit
221-SWITCHinfo says goodbye to
you.
221 You transferred 256 KBytes
during this session.
rif8%
```

Falls nicht reine Textdateien kopiert wird, sollte mit dem Befehl bin auf binäre Übertragung geschaltet werden. Viele Dateien sind weltweit mehrfach vorhanden. Um nicht unnötig Bandweite zu verschwenden und Kosten zu verursachen, sollten solche Dateien von einem Ftp–Server aus der Nähe kopiert werden. Der Server nic.switch.ch ist in der Nähe und unterhält aktuelle Kopien (mirrors) vieler bekannter Server. Es lohnt sich immer, zuerst auf nic.switch.ch nachzusehen.

Archie

Archie ist die Meta–Ebene vom anonymem Ftp. Ein Archie–Server speichert die vollständigen Inhaltsverzeichnisse von mehreren hundert Ftp–Servern weltweit. Mit Archie können gezielt Dateien gesucht werden. Ist der genaue Dateiname und Speicherort bekannt, wird die Datei mittels Ftp kopiert.
Zu Archie gibt es unter X Windows eine Client–Software (xarchie). Als Archie–Server wählt man archie.switch.ch, weil er nah und aktuell ist. Als Suchart hat sich Teilstring–Suche ohne Beachtung der Gross–/Kleinschreibung bewährt. Archie findet ein und dieselbe Datei meistens auf mehreren Ftp–Servern. Es ist dann sehr gezielt möglich, einen Ftp–Server in der Nähe auszusuchen und die Datei von dort zu kopieren. Findet Archie eine Datei nicht, von der man den Namen ziemlich genau kennt, so ist sie wahrscheinlich nirgends auf dem Netz öffentlich zugänglich. Die Client–Software xarchie übernimmt übrigens auch gleich noch das Kopieren mit Ftp. Für Archie und alle weiter unten vorgestellten Dienste gilt übrigens, dass sie nur kurzzeitig eine Verbindung mit dem entfernten Rechner aufnehmen und nach getaner Arbeit sofort wieder beenden .

Gopher

Gopher ist ein hierarchisches Menu zu Internet–Diensten. Es hat allerdings keine echte Baumstruktur; denn zwischen den einzelnen Ästen und Bäumen gibt es viele Verbindungen und einige Äste und Blätter werden dynamisch erzeugt. Hinter einem Menupunkt kann sich ein weiteres Menu, eine Datei oder eine Datenbank–Anfrage (z.B. das Telefon–Verzeichnis der ETH) befinden. Eine Datenbank–Anfrage liefert wiederum eine Datei oder ein Menu.
Wenn eine Datei gewählt wird, die eine dem Programm bekannte Endung hat, so wird sie nicht nur auf den lokalen Rechner kopiert, sondern ein Hilfsprogramm kann noch etwas mit ihr tun. Auf diese Weise werden Bilder angezeigt, Audio–Dateien abgespielt usw. Ein guter Startpunkt für eine Reise auf dem Internet ist der ETH eigene Gopher–Server: gopher.ethz.ch. Die Meta–Ebene zu Gopher nennt sich Veronica, ist an verschiedenen Stellen auf dem Internet anzutreffen und meistens überlastet.
Gopher beherrscht neben dem Gopher–Protokoll auch noch Ftp. Es kann also gut sein, dass sich hinter einem Menupunkt in Wirklichkeit ein Verzeichnis oder eine Datei auf einem Ftp–Server versteckt. Weiter gibt es noch Gateways, die einen anderen Dienst über das Gopher–Protokoll verfügbar machen. Bei Gopher geht die Übersicht schnell verloren. Meist hat man keine Ahnung, von woher eine Datei oder ein Verzeichnis geladen wurde und ob sie eventuell nur dynamisch generiert wurden.
Unter Unix kenne ich keinen brauchbaren Client für Gopher. Dafür

ist TurboGopher für den Mac umso besser. Warum es sich aber nicht lohnt, Gopher direkt zu benutzen, steht im nächsten Kapitel.

World-Wide Web

World-Wide Web ist der genialste aller Dienste auf dem Internet. Eigentlich ist es ein weltweit verteiltes Hypertext-System. Doch es als interaktive, multimediale Schatzsuche zu bezeichnen, trifft die Sache besser. Jedes Dokument des Systems ist sauber gestaltet und kann sogar eingebettete Bilder haben. Jeder beliebige Teil, also ein Bild, kann ein Link sein, der an eine andere Stelle im System führt. Diese Stelle kann noch im selben Dokument, aber auch am anderen Ende der Welt sein. Seit der Version 2 von WWW können die Hypertext-Dokumente auch wie Formulare aufgebaut sein. Die zur Verfügung stehenden Elemente wie Felder, Listboxen, Popup-Menu usw. stellen manchen Interface-Builder in den Schatten. Es können Audio-Dateien abgespielt und MPEG-komprimierte Filme angesehen werden.
Der mit Abstand beste Client für WWW heisst Mosaic und ist unter Unix, Windows und auf dem Mac verfügbar. Es ist ein ausserordentlich benutzerfreundliches Programm. In Mosaic werden Links blau dargestellt. Führt der Link an eine Stelle, wo man schon einmal gewesen ist, so wird er violett dargestellt. Zeigt man mit dem Mauspfeil auf einen Link, so wird der dazugehörige URL (Universal Resource Locator) angezeigt, also die Adresse, wo der Link hinführt. Damit wird schnell ersichtlich, woher das nächste Dokument geladen wird. Die Mosaic-Installation auf dem Studenten-Cluster ist auf dem neusten Stand und geradezu mustergültig. Ausnahmsweise geht damit ein grosses Lob an die Stabstelle Software. (Leif Hedstrom von der Stabsoft scheint ein angefressener Net-Surfer zu sein.)
Ein URL kann so aussehen: `http://www.ethz.ch/Index.html`. Der erste Teil gibt das Protokoll an, der zweite ist die Internet Adresse und der dritte ist eine genauere Bestimmung, in der Regel ein Dateiname. Auch an der ETH wurde gerade damit begonnen, einen WWW-Server aufzubauen. Sein URL ist `http://www.ethz.ch/`. Und der VIS hat ebenfalls einen Server: `http://vis-next.iiic.ethz.ch/`. Als Ausgangspunkt für einige sehr interessante Seiten kann ich `http://mtv.com/favlinks.html` empfehlen.
Mosaic beherrscht neben dem WWW-Protokoll, das sich HTTP nennet, auch noch Gopher, Ftp, WAIS, News und benützt rege Gateways. Damit ist Mosaic gleich auch noch ein Gopher-Client (siehe oben). Obwohl bei Mosaic eher als bei Gopher klar ist, woher das aktuelle Dokument stammt, gibt es hier den tückischen Fall, wo ein Dokument zwar von einem Server in der Nähe geladen wird, die eingebetteten Bilder

aber von Australien (oder von Deutschland, was noch langsamer ist). Hinzu kommt noch, dass wegen der vielen Bilder und Bildchen die kopierten Datenmengen nicht gerade klein sind. Es ist vor allem schwierig, einen MPEG–Film von weniger als einem MByte zu finden.
Wer auf dem Netz kommerzielle Software oder pornographische Bilder sucht, wird kaum etwas finden. Abgesehen davon, dass das meistens illegal ist, passiert immer das gleiche, wenn der Ort solcher Dateien bekannt wird: Der Server wird unablässig belagert von Leuten, welche die Dateien kopieren wollen, so dass innerhalb von Stunden der Server vom Netz abgehängt werden muss, bis sich wieder alle beruhigt haben. Danach sind die gesuchten Dateien nicht mehr auf dem Server zu finden.
Zu all diesen Internet–Diensten gäbe es noch viel zu sagen. Wer mehr wissen will, kann sich entweder eines der vielen Bücher über das Internet kaufen oder auf dem Internet selbst nach Dokumentation suchen. Dort findet man allein über diese vier Dienste so um die 500 Seiten Dokumentation. Ein Buch der ETH–Bibliothek zu diesem Thema auszuleihen ist keine Alternative. Denn obwohl das ETHICS einige Bücher zu diesem Thema führt, sind die Warteschlangrn enorm lang. Ob es in der Informatik–Bibliothek dazu Bücher gibt, weiss ich nicht; verfügbar sind sie bestimmt nicht, da die Bibliothek bekanntlich nur das antiquarische Bücher–Endlager des Departements ist.

Es wäre schön, wenn wir Studenten auch in Zukunft freien Zugang zum Internet hätten. Dazu muss aber jeder Student die vorgestellten Dienste verantwortungsvoll benutzen und darf nicht unnötig viel und unnötig weit her megabyteweise Daten kopieren.

Manuel Bleichenbacher IIIC/8
mbleiche@iiic.ethz.ch

Neue Skripte

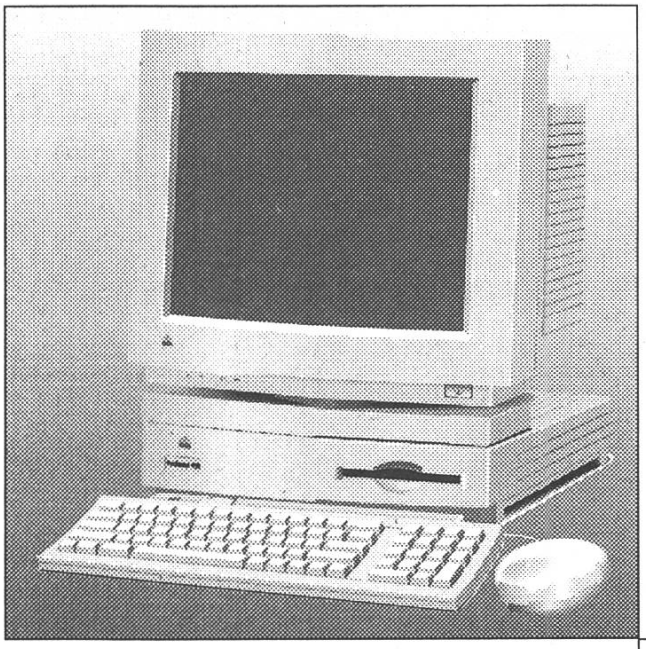
Im VIS Büro liegen zwei neue Skripte zu den Vorlesungen Informatik III (Prof. Mössenböck) und Informatik IV (Prof. Sanders). Das Skript zu Info III ist von Prof. Mössenböck persönlich, das zu Info IV ist von Prof. Sanders gebilligt.
Ihr könnt Euch diese Skripten zu den Präsenzzeiten gegen Hinterlassen der Legi zum Kopieren ausleihen.
lj

COMPUTER-LADEN

Computerladen
Rötelstrasse 135
CH-8037 Zürich
TakeAway
Tel. 01/362 72 90
HighEnd und Institute
Tel. 01/362 76 35
Fax 01/362 75 21
AL CH0024
Apple Education Center
Autorisierter
Apple Fachhändler

KOMPLETT? VERRÜCKT!

Ab Lager im COMPUTERLADEN! Ein komplettes System zum verrückten Preis!



PROMO 1

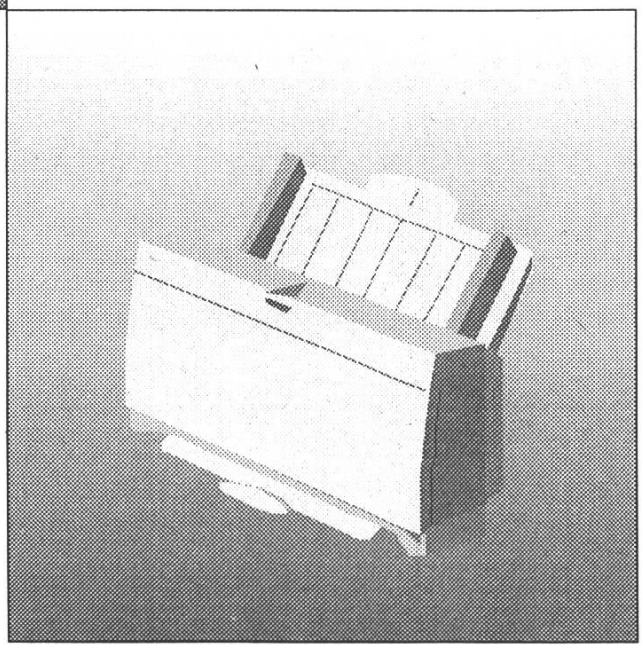
- **Performa 450:**
  4MB RAM 120MB Harddisk
  Works, PC Exchange, AtEase
- **14" Plus Display**
- **StyleWriter II**

Fr 1'940.-

PROMO 2

- **Performa 450:**
  4MB RAM 120MB Harddisk
  Works, PC Exchange, AtEase
- **14" RGB Display**
- **StyleWriter II**

Fr 2'470.-



Stiftung Zentralstelle der Studentenschaft der Universität Zürich
Eine Non-Profit-Organisation der Studentinnen und Studenten der Universität Zürich

Solange Vorrat!

Prüfungsergebnisse der Session Frühling 94

1. Vordiplom

	H 93	*F 94*
Anzahl Kandidaten	119	32
davon Repetenten	14	28
Bestanden	71 = 60 %	21= 65 %
davon Repetenten	4 = 9 %	19= 67 %
Durchschnitt	4.16	3.98
Standardabweichung	0.75	k. A.

Fächerstatistik	*Schnitt*	*Std'abw.*	*Durchf'quote*
Analysis I+II	4.21	0.92	29 %
Informatik I+II	4.06	1.11	39 %
Algebra I+II	3.78	0.86	45 %
Elektrotechnik I+II	3.93	0.79	35 %
W'keit & Statistik	3.74	0.78	52 %

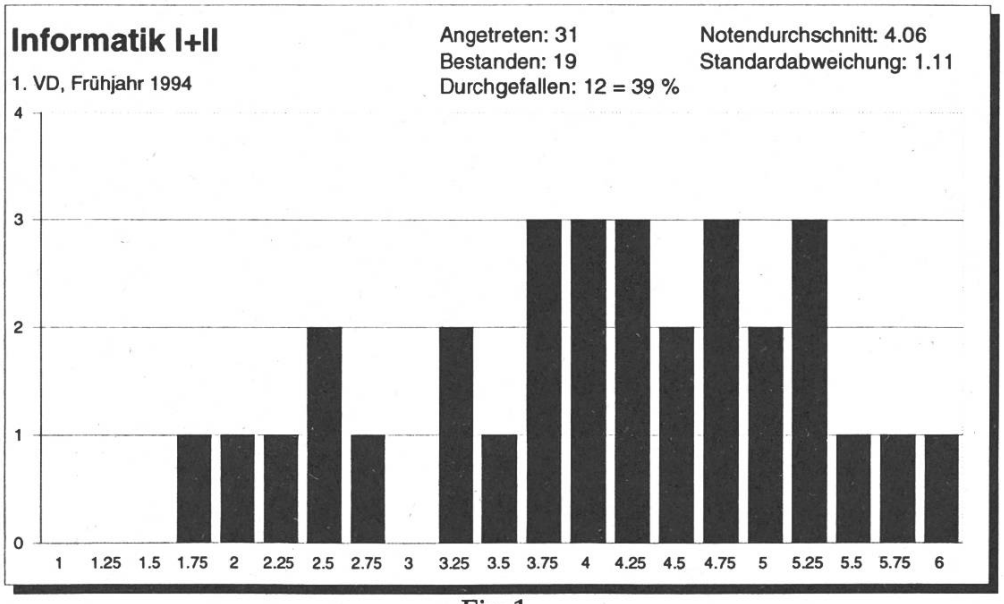


Fig.1

Das erste Vordiplom hat seine Zahlen gegenüber dem Herbst ein wenig verbessert, die Repetenten haben endlich einmal eine "gute" Durchfallquote. (Das heisst, dass sie niedriger ist, als die Gesamtdurchfallquote...) Sie haben entsprechend ihrem Anteil von 88 % die Gesamtdurchfallquote ziemlich stark beeinflusst. Da soll noch einer sagen: "Wenn mehr Repetenten zu den Prüfungen antreten, steigen die Durchfallquoten." Im Gegenteil: gefallen sind sie, auf eine dem 1. VD angemessene Marke. (Ob das so bleibt?)

Generell sind die oben gegeben Werte repräsentativ für ihre Verteilung. Dennoch möchte ich auf zwei Sachen hinweisen:

In der Informatik fällt die hohe Standardabweichung auf, was mich veranlasst, Euch die Verteilung in diesem Fach zu präsentieren (Fig. 1)

Diese Kurve streut auf beide Seiten ungefähr gleichmässig, ist aber relativ breit. Die Informatik ist zwar schon mit dreifachen Notengewicht ausgestattet, eine breitere Verteilungskurve nimmt aber noch mehr Einfluss auf die Endnote. Denn mit einer 1.75 in Informatik muss man schon in den Fächern Analysis, Algebra und Elektrotechnik mindestens eine 5.0 haben, dann darf man sich in Wahrscheinlichkeitsrechnung und Statistik sogar eine 4.75 leisten. Ein ziemlich hoffnungsloses Unterfangen, wenn man sich die Kurve in dieser Prüfung ansieht (Fig. 2)

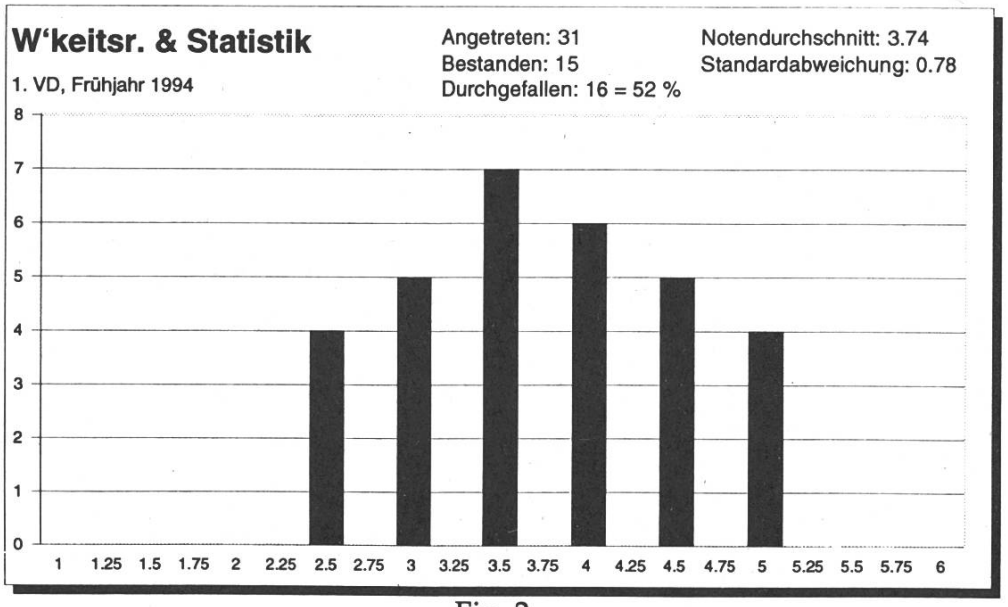


Fig. 2

Wie wir sehen, gab es in Wahrscheinlichkeitsrechnung und Statistik *keine* Viertelnoten. Unser 1.75–Informatiker musste also wohl oder übel auch in der W&S–Prüfung eine 5.0 hinlegen. Irgendwie glaub ich nicht ganz, dass er das geschafft hat. Der Fairheit halber möchte ich aber erwähnen, dass man sich mit einer 6.0 in Informatik in Analysis, Algebra und Elektrotechnik eine 3.25 leisten darf, in W&S sogar eine 2.5, um einen Gesamtschnitt von 4.0 zu erreichen. Die Informatik ist eben unsere Studienrichtung und dementsprechend hohes Gewicht muss sie im 1. VD haben. Die Tatsache, dass die Kurve in der Informatik I+II enorm breit ist, scheint mir aber auszudrücken, dass man mit dem dreifachen Notengewicht noch nicht zufrieden ist und mit der Streuung noch mehr Einfluss auf die Gesamtstatistik ausüben will ...

Die W&S–Prüfung bildet mit allen Werten ausser der Standardabweichung das Schlusslicht. Daran ist wohl vor allem die verheerende Vorlesung schuld. Hoffentlich wird das in diesem Semester besser. Schliesslich sind Kenntnisse in Wahrscheinlichkeitsrechnung und Statistik für einen Informatiker ziemlich wichtig. Ich wäre beispielsweise über bessere Kenntnisse in W&S froh, gerade, wenn es um die Bewertung der Prüfungsstatistiken geht. Es ist beschämend, dass jede Session so viele Leute aufgrund der schlechten Vorlesungen in diesem Fach durchfallen.

2. Vordiplom

	H 93	*F 94*
Anzahl Kandidaten	105	41
davon Repetenten	20	15
Bestanden	66 = 65 %	24 = 58 %
davon Repetenten	6 = 30 %	11 = 73 %
Durchschnitt	4.16	3.84
Standardabweichung	0.88	k. A.

Fächerstatistik	*Schnitt*	*Std'abw.*	*Durchf'quote*
Elektrotechnik III+IV	4.04	0.76	36 %
Wi. Rechnen I+II	4.04	0.82	38 %
Th. Informatik I+II	3.99	1.12	51 %
Informatik III+IV	3.58	1.05	62 %
Physik I+II	3.54	0.62	72 %

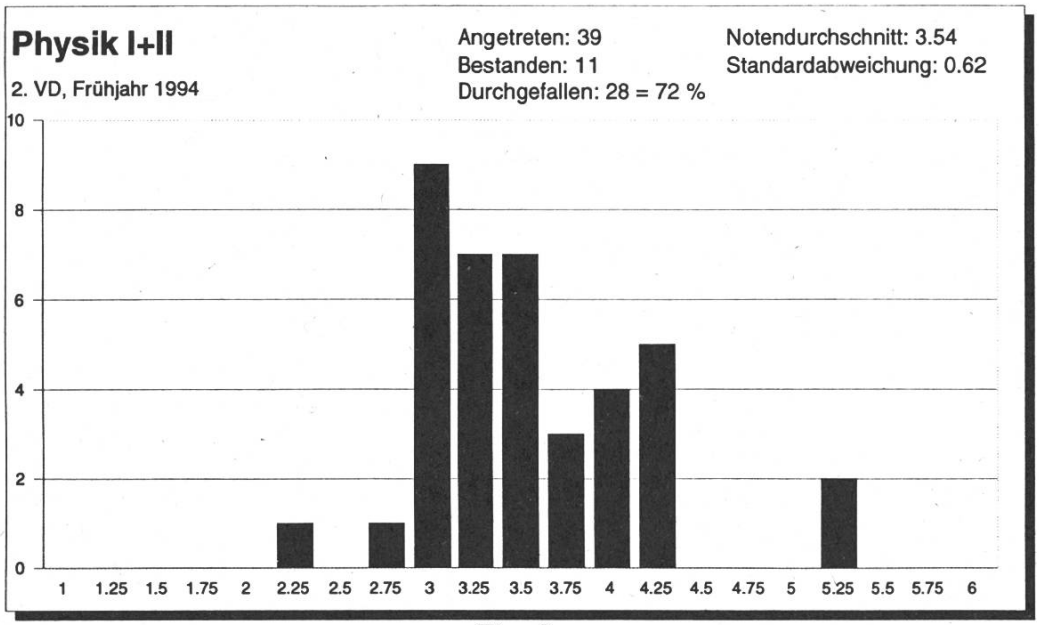


Fig. 3

In dieser Prüfung hat sich die Gesamtstatistik wieder einmal verschlechtert, und daran sind offenbar die Repetenten nicht schuld: Sie haben eine Durchfallquote von 27 %. Wieder einmal hat sich die Gesamtdurchfallquote in Richtung 50 % bewegt. Man mag sagen, dass diese Zahl nicht sehr aussagekräftig ist, dennoch ist sie höher als die 1. VD–Durchfallquote, und das ist sehr bedauerlich.

Drei Fächer haben eine Durchfallquote von über 50 %. Während sich die Theoretische Informatik noch halbwegs in der Nähe der Gesamtdurchfallquote befindet und einen gerade ungenügenden Schnitt aufweist, heben sich die Informatik III+IV und die Physik I+II im Schnitt schon recht deutlich vom Feld ab. Interessant ist, dass die beiden Fächer sowohl im Schnitt als auch in der Durchfallquote recht ähnliche Werte haben. Sehen wir uns also die beiden Histogramme an: (Fig. 3 & 4)

Eine Durchfallquote von 72 % in einer Prüfung ist sehr schlimm. Aufgrund solcher Ergebnisse kann ich nur noch den Survival Guide zitieren: "Keine Angst vor der Physik. Wenn sie in die Hosen geht, ist das nicht so schlimm." Was soll man denn anderes raten? Eine solche Verteilung führt uns doch das Glücksspiel Physikprüfung noch deutlicher vor Augen, und da diese Prüfung nicht viel mit den übrigen Fächern gemeinsam hat, sollte man sie wirklich nicht so ernst nehmen. Nach Augenzeugenberichten war die Prüfung in dieser Session (leider haben wir noch kein Exemplar davon) um einiges schwerer als die Herbstprüfung.

Die ungeheure Stoffülle und die Geschwindigkeit, mit der die Physik vorgetragen werden muss, trägt das ihre zum Nichtverständnis bei. Da hilft es auch nichts, wenn Prof. Leisis Vorlesung in der letzten Umfrage so gut abgeschnitten hat.

Ein Trost in der Physik war, dass die Verteilung (bis auf 2 Ausnahmen) nach unten nur bis 3.0 streut. Meines Erachtens ein Zeichen dafür, dass die Prüfung relativ wohlwollend korrigiert wurde. Da hatte man wenigstens eine Chance, diese Noten in den anderen Fächern noch ausgleichen.

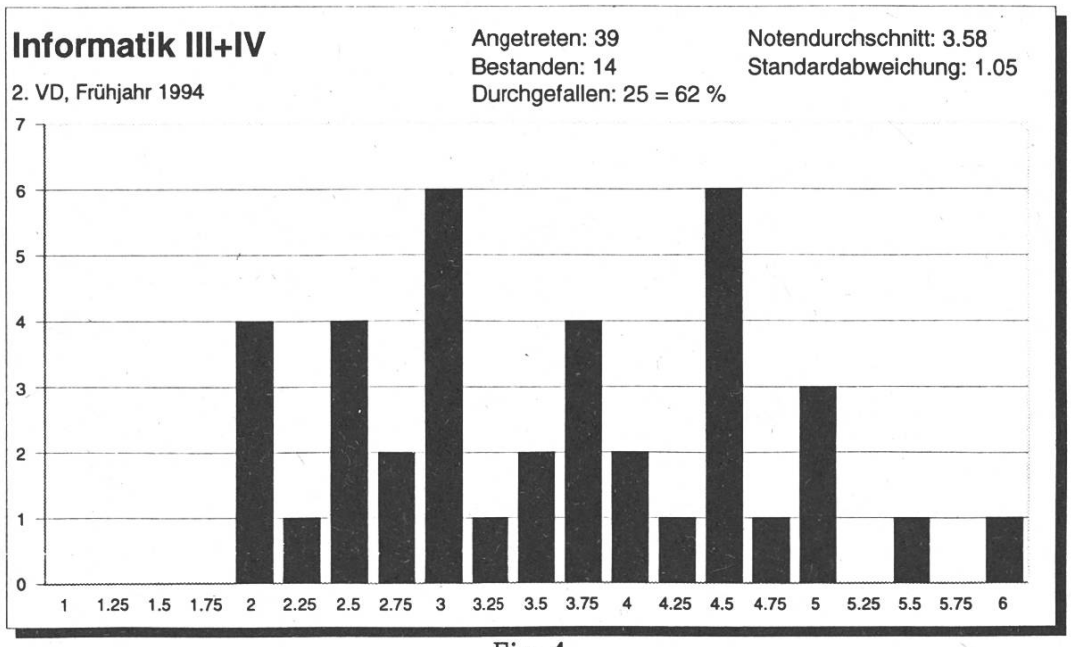


Fig. 4

Der Abteilungsvorsteher hat letzthin erwähnt, dass der Schnitt in Physik I+II wieder einmal bei 3.5 liege und dass das offensichtlich ein Zeichen dafür ist, dass die hohen Erwartungen, die man in der Physik stelle, nicht erfüllt werden können. Nun, in der Informatik III+IV haben wir einen Schnitt der nur um 0.04 Notenwerte besser ist als der Physikschnitt. Wenn wir hier das Histogramm betrachten, dann sehen wir schnell, dass die Informatik eigentlich der grosse Buhmann dieser Session ist: (Fig. 4)

Was soll man da noch sagen? Die Kurve streut nach unten bis zur 2.0. Der Unterschied zwischen 62 % und 72 % sind 3 Leute, dafür haben die anderen es schwerer, ihre Note auszugleichen. Kann man bei der Physik noch argumentieren, dass sie im allgemeinen für den Informatiker nicht so wichtig ist, so kann man das bei der Informatik III+IV wohl kaum behaupten. Dennoch überzeugt uns eine kleine Rechnung, dass mindestens 8 Leute[1], das sind immerhin 20 %, die nun neu ins Fachstudium eingetreten sind, un–genügende Kenntnisse in Informatik III+IV aufweisen. Wenn wir dieser Statistik glauben können. Aber es ist zu bezweifeln, dass diese Prüfung dazu gedient hat, Informatikstudierenden Kenntnisse in *Assembler–programmierung* und *Programmierung im Grossen* nachzuweisen: Wenn das Gerücht stimmt, dass bei der Korrektur dieser Prüfung die Punkte nach dem Alles–oder–Nichts–Prinzip vergeben werden, dann wird uns die hohe Streuung relativ klar.

Ausserdem sagt die Note nicht mehr viel aus: Ich kann schliesslich den Stoff verstanden haben, aber (beispielsweise aufgrund der Aufregung) mache ich bei der Prüfung Flüchtigkeitsfehler, und da ich nur die volle Punkteanzahl bekomme, wenn mein Programm vollständig stimmt, kann es unter Umständen vorkommen, dass ich überhaupt keine Punkte erhalte (aufgrund der Flüchtigkeitsfehler!). Es geht anscheinend bei einer solchen Prüfung nicht darum, Konzepte und Methoden zu kennen, sondern man muss sich unter Stress verhalten wie ein Roboter. Ich kenne einige Leute, die bei der Prüfung durchgefallen sind, und sie hatten kein so schlechtes Gefühl in Informatik III+IV. Sie hatten auch gedacht, den Stoff verstanden zu haben. Nun, das heisst ja nichts, nicht wahr? Aber irgendwie müssen diese Leute zu diesem Gefühl gekommen sein (beispielsweise, indem sie ihre Übungsprogramme zum Laufen gebracht haben — was mir allerdings zeigt, dass sie den Stoff wohl doch ein wenig verstanden haben). Wo bleibt da die Aussagekraft dieser Prüfung?

[1] 25 (in der Informatik durchgefallen)-17(insgesamt durchgefallen)=8

Schlussdiplom 1. Teil, Reglement '89

	H 93	*F 94*	
Anzahl Kandidaten	68	35	
davon Repetenten	k. A.	k. A.	
Bestanden	62 = 91 %	k. A. = 89 %	
davon Repetenten	k. A.	k. A.	
Durchschnitt	4.68	4.58	
Standardabweichung	k. A	k. A.	
Fächerstatistik	*Schnitt*	*Std'abw.*	*Durchf'quote*
Rekursivität & K.	4.89	0.77	17 %
Digitaltechnik & Rs.	4.57	0.64	11 %
Informationssysteme	4.44	0.69	25 %
System-Software	4.43	0.84	31 %

Am Schlussdiplom (nach dem alten Reglement) hat sich offenbar nicht viel verändert bis auf die Tatsache, dass die Durchfallquoten gestiegen sind. Meinetwegen mögen sie nicht sehr viel aussagen, aber eine Durchfallquote von 31 % in System–Software ist entschieden zu hoch, und auch die 25 % in Informationssysteme wollen mir nicht gefallen.

Dass es auch anders geht, zeigen die Prüfungen nach dem neuen Reglement. Mit der Gegenüberstellung der Prüfungen von Prof. Gutknecht und Prof. Mössenböck vermische ich zwar die Reglemente, sie dürfte aber viele Studierende interessieren: (Fig.5)

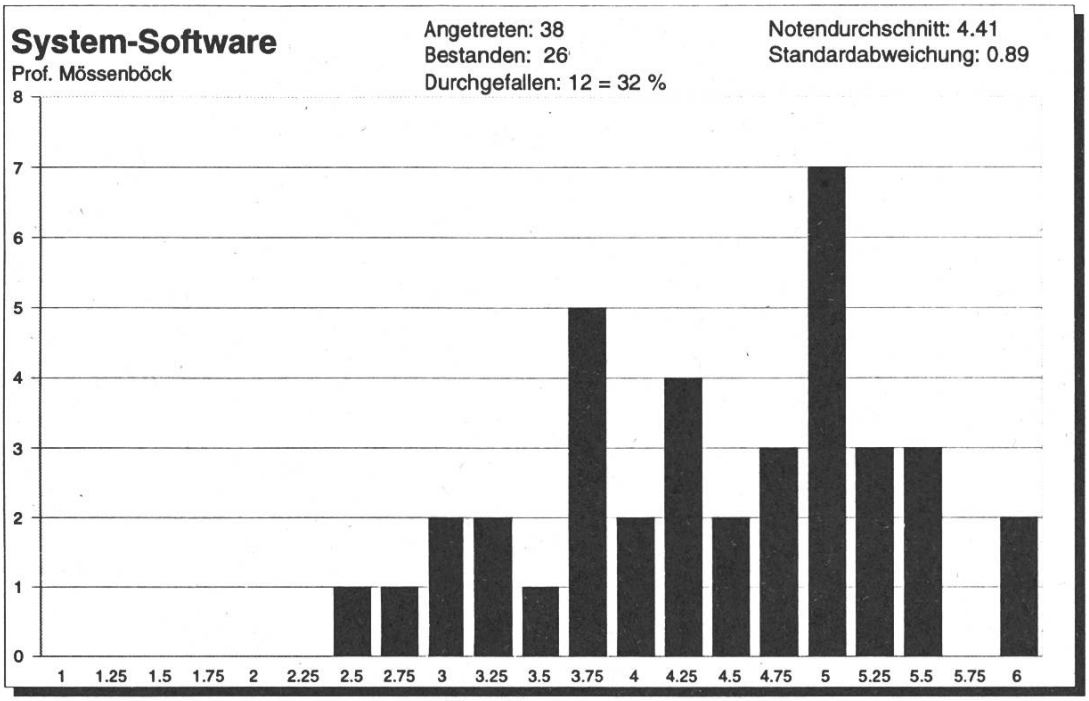


Fig. 5

Wenn Prof. Mössenböck eine solche Durchfallquote in Informatik III im Herbst erreicht, bin ich mit ihm zufrieden. Aber irgendwie glaub ich nicht ganz, dass es so sein wird.

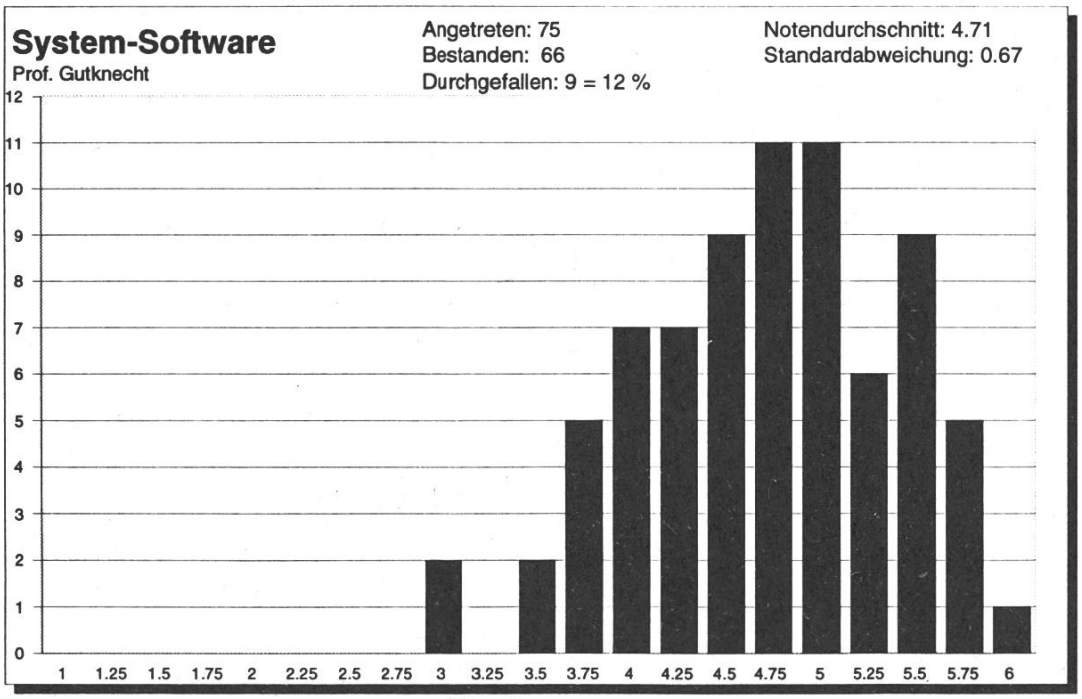


Fig. 6

Schlussdiplom 2. Teil, Reglement '89

Alle 20 Kandidaten haben den 2. Teil der Schlussdiplomprüfungen erfolgreich abgeschlossen. Ausserdem haben 74 Studierende, darunter auch eine Frau, mit Erfolg ihr Studium abgeschlossen und dürfen sich jetzt *Diplom-Informatik-Ingenieur(in)-ETH* nennen.
Drei weitere Kandidaten haben noch nach dem Reglement 1981 ihr Studium erfolgreich beendet.

Kernfachprüfungen im Fachstudium, Reglement '93

Fächerstatistik	*Anzahl Kand.*	*Schnitt*	*Std'abw.*	*Durchf'quote*
Digitaltechnik & Rs.	16	4.94	0.63	0 %
Rekursivität & K.	18	4.86	0.74	6 %
Informationssysteme	91	4.79	0.54	5 %
System-Software	87	4.71	0.75	13 %

Hier präsentiere ich Euch nur das Diagramm in Informationssysteme. In Systemsoftware sieht es dem von Prof. Gutknecht relativ ähnlich. Viel gibt es dazu nicht zu sagen, denn die Prüfungsergebnisse sind sehr erfreulich. Die 5 % in Informationssysteme sind endlich einmal eine Durchfallquote, die man gut nach aussen hin vertreten kann. Und auch die 13 % in System-Software sind ok. (Fig. 6)

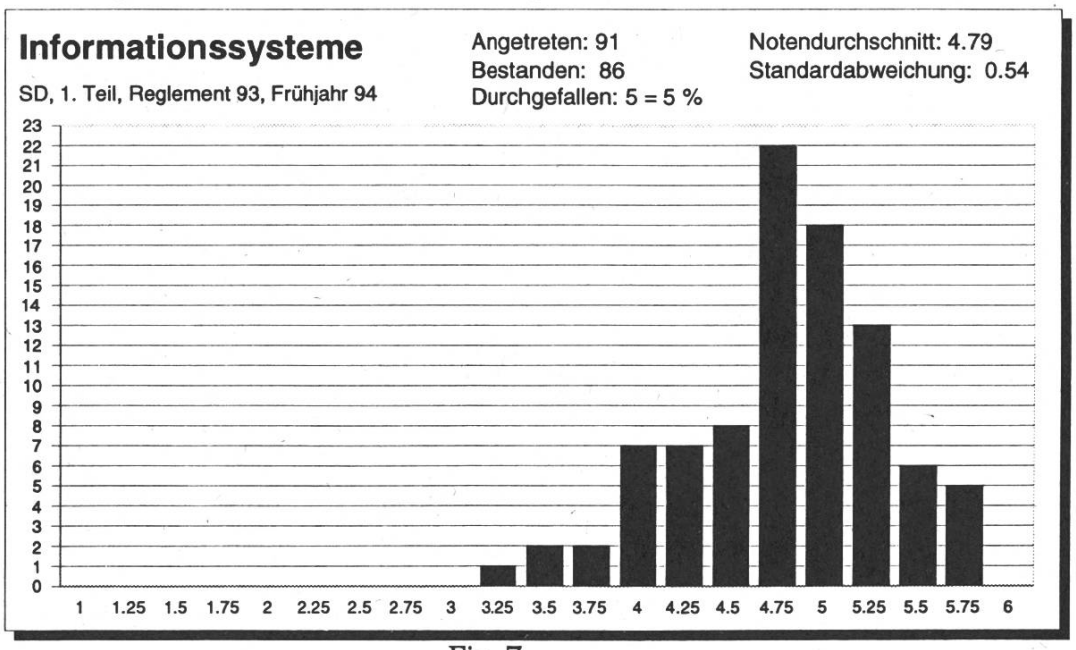


Fig. 7

Ich gratuliere allen recht herzlich, die die Prüfungen (sei es nun 1. VD, 2. VD oder SD, Teil x) bestanden haben. Weil ich bei der nächsten MV mein Ressort abgebe, habe ich da noch ein Nachwort:

Ceterum censeo ...

Die hohen Durchfallquoten gereichen uns nicht (nur) zur Ehre: Manche Professoren meinen zwar, dass hohe Durchfallquoten dem Ruf einer strengen Schule gerecht werden, und dass damit grosse Qualität bekundet wird.

Aber genauso wird jeder mathematisch gebildete Mensch einsehen, dass bei einer gewissen Quote der Zufall regiert. Mit anderen Worten: Es existiert eine Notenverteilung, bei der die Qualität der Vorlesung und ihre Prüfungsergebnisse nicht mehr korreliert sind. Aus der Annahme, dass die Mehrheit der Studierenden einer Vorlesung vernünftig ist, müsste sich eigentlich folgern lassen, dass der Grossteil auf die Prüfung lernt und diese dann auch besteht, dass also die Durchfallquote statistisch relevant unter der 50–%–Marke liegt. Je näher die Quote bei 50 % liegt, umso mehr sind die Vorgänge vor und während der Prüfung nicht mehr korreliert: D.h. die Vorlesung entfernt sich mehr und mehr vom Prüfungsstoff. In diesem Fall verliert die Voraussage "Wer viel und gut lernt, kommt bei einer Prüfung durch" ihren Wahrheitswert. Praktisch ist keine Voraussage mehr möglich – die Prüfung wird zum Glücksspiel. Bei einem Steigen der Durchfallquote über 50 % wird die Korrelation zwischen Prüfung und Vorlesung negativ, d.h. es müsste vermehrt das gelernt werden, was in der Vorlesung nicht vermittelt werden konnte[2]. Eine Durchfallquote im Bereich 50 % ist also eine gefährliche Sache: Sie lässt im Zusammenhang mit Vorlesungsumfragen (die diese These unterstützen) auf schlechte Qualität von Vorlesungen schliessen.

Oder die Studierenden sind unvernünftig. Auch in dieser Hinsicht leiden die Studierenden (wie auch in der Folge die Absolventen) unter den bestehenden Zuständen: Eine Beobachtung bei Leuten anderer Fachrichtungen, die über die hohen Durchfallquoten an der ETH–Informatik informiert sind, zeigt, dass sich zumindest in einigen dieser Köpfe eine recht irrationale Meinung über Informatiker gebildet hat. Nämlich die, dass die Informatiker schon ein recht seltsames, un–intelligentes Völkchen sein müsssen, wenn mit einer derartigen Konstanz jede Session solche Durchfallquoten produziert werden. Diese Meinung bezieht sich dann natürlich nicht nur auf die

[2]Wer diesen Denkschritt nicht mitmachen kann, der überlege sich doch einfach den härtesten Fall einer Durchfallquote von 100 %: Wenn alle Studenten, die zu einer Prüfung angetreten sind, durchfallen, dann muss man annehmen, dass niemand von ihnen die Vorlesung verstanden hat. Denn aus der Annahme, dass ein Grossteil der Studenten vernünftig ist, muss ich folgern, dass sie auf die Prüfung lernen.

Durchgefallenen, nein, sie weitet sich auf alle Informatiker aus. Kurz: Studierende der anderen Abteilungen an der ETH haben von uns Informatikern im allgemeinen keine gute Meinung, und es ist nur eine Frage der Zeit, bis diese Meinung beispielsweise auch in die Wirtschaft gelangt.

Sollte wirklich das unter Studierenden kreisende Gerücht, dass die VD–Durchfallquoten an der Informatik künstlich hoch gehalten werden, eine kleine Wahrheit enthalten, so müsste ich den betreffenden DozentInnen vorwerfen, dass sie ihren eigenen Nachwuchs dezimieren. Die hohen Durchfallquoten haben immerhin schon im Fachstudium zu einer erheblichen Verkleinerung der Studierendenzahlen geführt, und es ist zu bezweifeln, dass das relativ grosse Angebot an Vertiefungs–vorlesungen gehalten werden kann, wenn die Zahlen in Zukunft noch schrumpfen. Vielleicht kann ja der neu geschaffene "Nachdiplomkurs", der die Vertiefungsvorlesungen füllen soll, die Abteilung aus diesem Dilemma führen.

Die hohen Durchfallquoten schrecken natürlich Neueintretende ab: Auch im Grundstudium beginnen die Studentenzahlen kleiner zu werden. Letzthin wurde mir gegenüber die Meinung geäussert, dass die hohen Durchfallquoten die Leute vom Informatikstudium abhalten, die die Vordiplome ohnehin nicht schaffen würden. Deshalb trete nun auch eine kleine Erleichterung in den Durchfallquoten ein. Aber vielleicht sollte man sich auch überlegen, dass viele begabte Maturanden nicht unbedingt das Selbstvertrauen haben, das man von ihnen hier offenbar erwartet, und dass diese jungen Leute, obwohl sie es schaffen würden – entsetzt von den hohen Durchfallquoten – nicht den Mut haben, vier Jahre ihres Lebens zu riskieren, um danach vor dem Nichts zu stehen. Ich denke, dass der Abteilung (und in der Folge auch dem Departement) durch diese Abschreckung ein grosser Teil von potentiell hervorragenden Informatikern an die HTLs, an die UNI Zürich oder gar an andere Studienzweige verloren geht.

Nun habe ich – bis auf ein Argument – keinerlei Wortmeldungen der Befürworter hoher Durchfallquoten an–geführt. In den vier Jahren, die ich nun an der ETH Informatik studiere, habe ich aber auch nie irgendeinen anderes Votum für hohe Durchfallquoten gehört, als das, welches ich an den Anfang dieses Abschnittes gestellt habe. Und doch hat es den Anschein, als erschlage dieses Argument alle anderen. Eigentlich schade, habe ich doch den Eindruck, dass es genügend Argumente[3] auf der anderen Seite gibt, die dieses nahezu "geflügelte Wort" entkräften können.

Leonhard Jaschke

[3]Man findet in älteren Visionen noch andere, ich habe hier beileibe nicht alle Argumente gegen hohe Durchfallquoten aufgezählt.

C-Kurs 4. (und vorläufig letzter) Teil

Manche Leute füllen mit diesem Thema ganze Bücher, wir unsere Visionen. Aber vielleicht kann man mit den 4 Folgen unseres C-Kurses auch schon ein Buch füllen (so lang, wie sie manchmal waren). Wir überlegen auf jeden Fall, ob wir die Folgen in einem eigenen Handout zusammenfassen und dieses veröffentlichen sollen. Es kommt halt darauf an, wie gross das Interesse daran ist.

In dieser Folge müssen wir noch Themen aufgreifen, die wir bisher vernachlässigt haben. Eines davon ist

Makros

Makros sind einfache Ausdrücke, die der Präprozessor anstelle ihres Namens textuell vor der Kompilation ersetzt. Definiert werden die Makros deshalb mit einem Präprozessorbefehl: #define. Die einfachsten Makros haben wir schon kennengelernt: Konstanten. Die Zeile

```
#define PI
3.1415926535879323846
```

definiert die Konstante PI, die vor der Kompilation überall im Text vom Präprozessor durch ihren Wert ersetzt werden. Nun können wir dieses Makro auch variabel machen, in dem wir PI mit einer Unbekannten X multiplizieren. Dann sieht die Makrodefinition so aus:

```
#define PI(X)
(X*3.1415926535879323846)
```

wenn wir jetzt im Quelltext PI(3) schreiben, dann ersetzt der Präprozessor diesen Ausdruck durch 3*3.1415926535879323846. Zugegeben dieses Makro ist nicht sehr sinnvoll. Aber beispielsweise verwenden manche C-Programmierer das folgende Makro ganz gerne:

```
#define SQR(X) ((X)*(X))
```

Man beachte die Klammern, die ich um die X's gesetzt habe. Wären sie nicht vorhanden, hätte die folgende Quelltextzeile eine ungewollte Wirkung:

```
x= SQR(a+b);
```

Diese Zeile würde vom Präprozessor folgendermassen ersetzt: a+b*a+b, was nicht die intendierte Wirkung ist, lieber hätten wir eben (a+b)*(a+b) gehabt. Solche Sachen sollte man sich immer überlegen. In den Makros kann tierisch der Wurm drin sein.

Einige nützliche Makros zum Umwandeln von Strings in Zahlen möchte ich Euch nicht vorenthalten: Sie heissen atof, atoi, atol. Auf den meisten Unix-Rechnern sind sie binäre Funtionen (im Standard-C sind sie aber Makros) und als solche stelle ich sie Euch vor:

```
double atof(const char *s);
int atoi(const char* s);
long atol(const char *s);
```

Mit atof wandelt man einen ASCII-String in eine double (float) Zahl um, mit atoi und atol wandelt man diesen String dementsprechend in eine Ganzahl bzw. in eine lange Ganzzahl um.

Die andere Richtung möchte ich Euch auch nicht vorenthalten. In diese Richtung macht das eine *echte* Funktion. Aber diese zu finden, ist für einen Anfänger nicht leicht: Sie heisst *sprintf* und hat folgende Definition:

```
int sprintf(char *s, const
char *format, ...)
```

Bist auf den ersten Parameter kennen wir alle anderen von der Funktion *printf*. Der erste Parameter gibt den String an, in den das Resultat geschrieben werden soll. Ein Beispiel:

```
int zahl= 3;
sprintf(s,"%d",zahl);
```

Diese Zeile schreibt die Ganzzahl *zahl* (also ihren Inhalt 3), in den String *s*. Nach dieser Zeile hat *s* also den Wert "3".

Prozedurvariablen

Dieses wichtige Kapitel haben wir noch nicht besprochen, und es kann trotzdem durchaus einmal vorkommen, dass Ihr beispielsweise eine allgemeine Funktion schreiben wollt, die bestimmte Typen von Funktionen als Parameter nehmen kann. Um das zu tun, braucht man also Prozedurvariablen. Wie sehen sie aus? Eine Prozedurvariable ist einfach ein Pointer auf eine Funktion. Programmiert sieht das dann so aus:

```
float (*f)(float x);
```

Das definiert uns die Variable f als eine einstellige Funktion, die einen Parameter x vom Typ float nimmt und ein float-Resultat zurückliefert. Beispielsweise kann man der Variablen f nun die Sinusfunktion zuweisen:

```
f= sin;
```

Wenn wir jetzt f(3.1415) aufrufen erhalten wir sin(3.1415) ≈ 0.

Da der obige Ausdruck für eine Prozedurvariable relativ kompliziert ist, gebe ich Euch nun das Universalrezept in EBNF. Eure Funktion/Prozedur habe ich da so definiert:

```
FunctionDefinition ::=
FunctionType FunctionName "("
ParameterList ")".
FunctionType ::= Type.
Type ::= ["*"] BasicType |
UserDefinedType.
ParameterList ::= ["const"]
Type VariableName {","
["const"] Type VariableName}.
```

Dann definieren wir einen Typ für diese Funktion wie folgt:

```
FunctionTypeDefinition ::=
"typedef" FunctionTypeName
"(*" FunctionType ")("
ParameterList ");".
```

Eine Variable von diesem Typ wird ganz einfach mit

```
FunctionTypeVariableDefinition
::= FunctionTypeName
VariableName ";".
```

deklariert. Dieser Variablen könnt Ihr mit

```
FunctionVariableAssignment ::=
VariablenName "=" FunctionName
";" .
```

die obige Funktion zuweisen.

Fantastisch, nun können wir die Funktionen Variablen zuweisen, aber was soll das Ganze? Stellt Euch vor, Ihr

schreibt ein Programm wie Matlab, das mit Vektoren rechnet. Dieses Programm soll also relativ viele Funktionen auf Vektoren zur Verfügung stellen, beispielsweise Addition und Subtraktion von Vektoren, komponentenweise Multiplikation und Division, ... Wir können eine recht allgemeine Prozedur schreiben, die zwei n-dimensionale Vektoren a und b als Eingabeparameter nimmt, und jede Komponente von a mit der entsprechenden Komponente von b in einer Funktion f verknüpft. Die ganze Sauerei wird in einem Ausgabe-Vektor c gespeichert. Die Funktion f kann nun die Addition, Subtraktion, Multiplikation oder Division der beiden Komponenten sein. Da es da viele Möglichkeiten gibt, spezifizieren wir sie nicht, sondern verwenden f als eine Prozedurvariable. Aber seht selbst:

```
typedef float
(*funktion)(float x, float y);

float addieren(float x, float
y)
{
   return(x+y);
}

void verknuepf(float a[],
float b[], int n, funktion f,
float c[])
{
   int i;
   for(i= 0; i<n; i++)
   {

   c[i]= f(a[i],b[i]);
   }
}

void main(...)
{
float a[5],b[5],c[5];
...
verknuepf(a,b,5,addieren,c);
...
}
```

Getrennte Kompilation

Unter dieser Überschrift wird es nun Unix-spezifisch. Wir gehen hier nur auf C unter Unix ein, da wir an der ETH-Informatik vor allem auf den SUNs in C programmieren. Für andere Plattformen kann die getrennte Kompilation anders aussehen. Aber da habt Ihr dann (hoffentlich!) ein Handbuch, das Euch über diese Sachen Auskunft erteilen soll. (Überdies wird den Compilern auf anderen Plattformen oft ein dem Unix-Make[s.u.] ähnliches Make mitgeliefert).

Linking

Wie in Oberon kann man auch in C ein (grosses) Problem in kleinere Kompilationseinheiten unterteilen und diese getrennt kompilieren. Eine kurze Repetition an dieser Stelle: Wenn man Funktionen und Variablen NICHT exportieren möchte, dann kennzeichnet man diese mit dem Schlüsselwort "static" vor der Definition. Importiert werden Funktionen mit Hilfe des Schlüsselwortes "extern" vor der Definition: Man muss die importierten Element also beim Importieren noch einmal definieren (bei Variablen vollständig, bei Funktionen muss der Funktionskopf angegeben werden). Am besten macht man das, indem man die Schnittstelle der Kompilationseinheit in einem Header-File (.h) festlegt. Dieses entspricht dann in etwa dem Definition-Module von Modula (oder dem .sym-File von Oberon). Ein #include importiert dann alle Elemente aus der entsprechenden Kompilations-einheit.

Generell muss man beim Linken zwischen zwei Dingen unterscheiden: Unix kennt sowohl das "dynamische Linken" (mit dem wir in Oberon Bekanntschaft gemacht haben) als auch das "statische Linken". Beides ist mit ungefähr gleich viel mehr Aufwand verbunden als in Oberon. Widmen wir uns zuerst der in Unix im allgemeinen üblichen Variante:

Statisches Binden

Angenommen wir haben nur ein Programm "Prog.c" geschrieben. Wenn wir die Compileroption "-c" verwenden, wird das File NUR kompiliert. D.h. gcc erzeugt ein Objectfile "Prog.o", wenn wir ihn mit folgender Zeile aufrufen:

```
gcc -c Prog.c
```

Nun können wir mehrere Files (sagen wir "Prog1.c" und "Prog2.c") auf dieselbe Weise kompilieren. Wichtig ist dabei, dass genau ein File die Funktion "main" enthält! D.h. nur ein Modul darf unser Hauptprogramm enthalten! Für den Aufruf von Initialisationsprozeduren für die einzelnen Module ist also der Programmierer selbst zuständig (das geht nicht so komfortabel wie in Oberon). Das statische Linken erfolgt mit einem zweiten Aufruf von gcc, jetzt unter Angabe der Object-Files:

```
gcc -o Prog Prog.o Prog1.o
Prog2.o
```

Die Option "-o" haben wir schon in einer der früheren Folgen kennengelernt. Sie teilt dem Compiler mit, welchen Namen das ausführbare File erhalten soll. Nun, das ist nicht alles: Schliesslich wird uns mit Unix ein Haufen Betriebssystem-Bibliotheken mitgeliefert, die wir gerne auch benutzen würden. Beispielsweise könnte unser obiges Programm eine Library namens "curses" benötigen (eine Bibliothek zur Steuerung von Terminal-IO). Dann müssen wir diese Library mit Hilfe der Option -l (statisch) einbinden:

```
gcc -o Prog Prog.o Prog1.o
Prog2.o -lcurses
```

Das muss genauso zusammengeschrieben werden. Der (im gcc eingebaute) Linker sucht dann im LD_LIBRARY_PATH (Environment-Variable) nach dem File "libcurses.a" (a steht für archive) und linkt die benötigten Prozeduren hinzu.

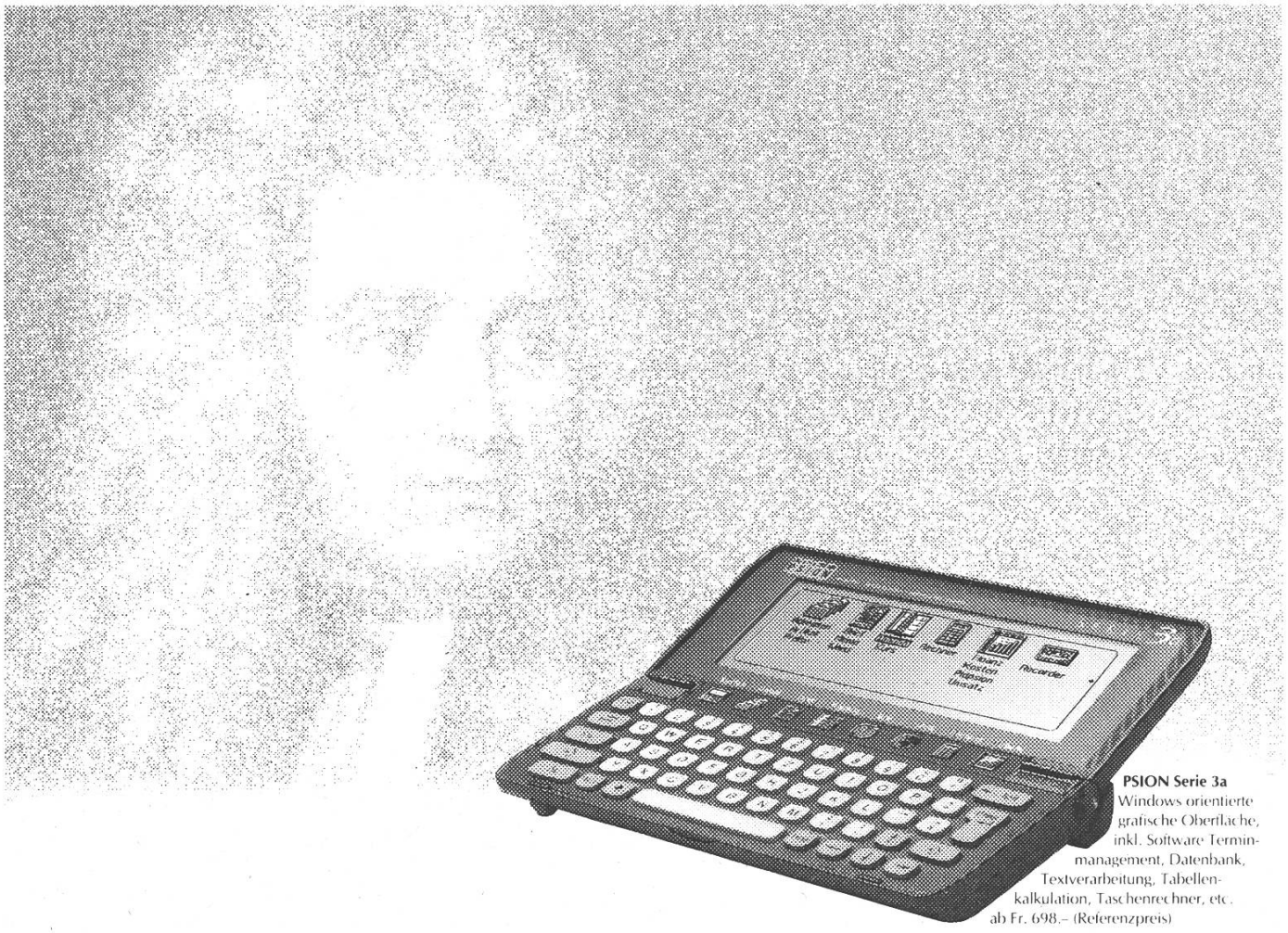
Im Falle mehrere Libraries kann es vorkommen, dass die eine die andere benötigt. In diesem Falle muss diese in der Optionenliste vor der benötigten kommen.

Beispiel: Wir benoetigen zum Linken unseres Programmes Prog (der Einfachheit haben wir nur ein Object-File Prog.o) die Libraries lib1, lib2, lib3. Lib1 benötigt lib2, lib3 benötigt lib1. Wie lautet der Aufruf?

```
gcc -o Prog Prog.o -llib3 -
llib1 -llib2
```

⇒ Seite 31

ADWORK



PSION Serie 3a
Windows orientierte grafische Oberfläche, inkl. Software Terminmanagement, Datenbank, Textverarbeitung, Tabellenkalkulation, Taschenrechner, etc.
ab Fr. 698.– (Referenzpreis)

Wenn der Herr Newton nur schon den PSION Serie 3a gehabt hätte!

Die Schwerkraft entdecken, das hätte er natürlich immer noch sollen. Aber dann wär's einfacher geworden. Ohne Griffel und Schiefertafel wäre die Information präzise und richtig über die praktische Tastatur eingegeben worden. Und er hätte seine Eintragungen auch morgen noch fehlerfrei entziffern können. Ohne Interpretationsspielraum in die integrierte Datenbank geswitcht und die neuen Naturgesetze grafisch dargestellt. Dann in das Schreiben an James Watt einkopiert. Und dank dem Multitasking des PSION Serie 3a gleichzeitig alles im besten Zeitplaner verwaltet, so dass er sich seinen Schreiberling glatt hätte sparen können - oder sogar für Entdeckungen einsetzen, von denen wir jetzt halt nichts wissen. Aber eben, wenn er nur gehabt hätte.

PSION - und Sie haben alles dabei!

Zum Glück sind Sie ja nicht Newton und können somit die Chance auf ein einfacheres und noch erfolgreicheres Leben ergreifen. Mit dem neuen PSION Serie 3a! Dank der hohen Bildauflösung und dem grossformatigen Bildschirm wird die enorme Computerleistung auch optimal dargestellt. Sei es bei grafischen Darstellungen der eingebauten Tabellenkalkulation. Beim Texten. Oder beim völlig neu konzipierten Terminplaner, der überzeugenden Alternative zu konventionellen Zeitplansystemen.

Die Intelligenz des PSION Serie 3a entlastet. Immer und überall. Und kostet erst noch viel weniger als eine nachgebaute Schiefertafel. Nur - Strichmännchen können Sie halt nicht zeichnen damit!

Da will ich mehr wissen - Bitte senden Sie mir die Unterlagen über den PSION Serie 3/3a:

Name _____

Adresse _____

PLZ, Ort _____

Generalvertretung für die Schweiz:
EXCOM AG, Moosacherstrasse 6, Au
8820 Wädenswil
Telefon 01/782 21 11, Fax 01/781 13 61

EXCOM
Ein Unternehmen der INFAREX Gruppe

Praktikumsbericht ABB Netcom AG

In diesen Frühlingsferien wollte ich den ersten Teil meines Industriepraktikums absolvieren. Nach einer Durchsicht der Liste der Praktikumsfirmen (einsehbar in EZINFO oder per ftp von neptune) informierte ich mich bei einigen Firmen zuerst telefonisch, ob sie überhaupt noch Praktikanten betreuen. Von den 7 Firmen, die mich interessierten, blieben danach noch zwei übrig. Eine weitere Möglichkeit bot sich an der Kontaktparty, so dass ich mich zwischen drei Angeboten zu entscheiden hatte. Nach einem Besuch der ABB Netcom AG in Gebenstorf entschied ich mich für diese Firma.

Arbeitsbeginn

Nach meiner Zusage erhielt ich meinen Arbeitsvertrag per Post. An meinem ersten Arbeitstag fuhr ich nach Turgi, da sich dort die Personalabteilung befindet. Von dort wurde ich von der Sekretärin nach Gebenstorf gefahren, wo die ABB Netcom AG (nach der Aufteilung der ABB sind ja die einzelnen Firmen rechtlich selbständig) in einem flachen Gebäude an der Reuss untergebracht ist.
Nach einem Rundgang durch die Firma durfte ich meinen Arbeitsplatz beziehen. Der bestand aus einem grossen Schreibtisch, dem nötigen Schreibmaterial, einem VT320–Terminal und wurde bald darauf um eine VAXstation 3100 und ein Telefon erweitert.

Projektbeschrieb

In einer Einarbeitungsphase konnte ich mich mit dem Produkt, an dem ich arbeitete, bekannt machen. Ich wurde beim PN40 eingesetzt, einem modularen System bestehend aus Hard– und Software für die Netzleittechnik. Ein solches System ermöglicht die Ueberwachung und Steuerung von Strom–, Gas–, Wasser– und ähnlichen Netzen. Speziell ist an der Software unter anderem, dass sie auf eine von der Firma selbst entwickelten Real–Time–Datenbank zurückgreift. Bei dieser Datenbank wird möglichst viel der für einen Datenbankzugriff nötigen Arbeit bereits in der Initialisierungsphase des Programms durchgeführt. Bei dieser Initialisierungsphase wird dann die Zugriffsstruktur vorbereitet, so dass die eigentlichen Zugriffe sehr rasch abgearbeitet werden können. Das bedingt natürlich, dass bereits im Voraus der Zugriffsmodus (Read / Write / Modify) und der Suchmodus (First / Last / Equal / Less...) festgelegt wird.

Nachdem mir die speziellen Eigenschaften dieser Datenbank einigermassen klar waren, konnte ich ein Programm zur Analyse einer bestehenden Datenbank entwerfen und realisieren. Dem Benutzer wird

damit die Möglichkeit gegeben, den Status einer Datenbankgenerierung zu überprüfen.

Die Softwarekomponente, an welcher ich arbeitete, war in VMS Pascal geschrieben, einer (zum Glück sehr) stark erweiterten Version des Wirth'schen Pascals. Als Entwicklungsstation konnte ich ein ALPHA mit OpenVMS verwenden. Dies gab mir Gelegenheit, mich mit VMS auseinander zu setzen. Von den Handbüchern gab es im Büro gleich mehrere Sets, so dass ich mir die interessanten gleich in Griffweite auf meinen Tisch stellen konnte.

Bei meiner zweiten Arbeit beschäftigte ich mich mit dem Calculations Editor, welcher dem Benutzer die Möglichkeit gibt, auszuführende Berechnungen (im Netzwerk) zu definieren. Das Ziel war es, die binär gespeicherten Daten so in ein File zu schreiben, dass dieses Benutzereingaben simulieren kann und zwar so, dass alle Daten neu erfasst werden. Dies macht es möglich, die Daten in handlichem (ASCII) Format zu speichern und Aenderungen einfach an diesem Eingabefile durchzuführen.

Die dritte Arbeit, die Anpassung des Calculation Editors an den heutigen "State of Art", führte zu einer totalen Ueberarbeitung der Programmstruktur. Unter anderem fasste ich mehrfach vorhandene Programmsegmente zusammen, welche dank 'Cut– and Paste'– Programmierstil entstanden waren, also nach dem Motto: So was ähnliches hatte ich schon, da kann ich ja einfach kopieren und etwas anpassen. BITTE – falls Ihr mal so einen Gedanken habt, geht doch den etwas mühsameren Weg und schreibt gleich gemeinsame Routinen. Jeder, der mal Euren Code warten muss, wird froh sein...

Kulinarisches und Sonstiges

In Gebenstorf gibt es keine Kantine. Dafür einen Spezialbus, der am Mittag nach Turgi ins Personalrestaurant fährt. Wie an der ETH ist dort der SV–Service für das Essen zuständig. Allerdings ist der Preis etwas höher (man verdient ja schliesslich) aber dafür auch die Bedienung nicht überlastet und äusserst freundlich. "Nach dem Essen sollst Du ruhn oder tausend Schritte tun". Dass es exakt tausend Schritte sind, bezweifle ich, aber guttuend ist so ein Spaziergang von Turgi zurück nach Gebenstorf auf jeden Fall. Der Weg führt nahe am Wasserschloss des Aargaus, dem Zusammenfluss von Aare, Reuss und Limmat, vorbei. Bei solchen Spaziergängen erfolgte immer ein reger Informationsaustausch, was sehr informativ war.

Sehr angenehm war, dass alle sich Zeit nahmen (selbst wenn sie noch so im Stress waren) auf meine Fragen zu antworten und auch sehr offen über heiklere Themen mit mir sprachen.

Im Büro gab es (fast) immer frischen Kaffee, den man per Strich hinter seinem Namen registrierte und wöchentlich zu 80 Rp. pro Strich bezahlte. So ein Strichsystem ist sehr interessant – man weiss endlich wieviel Kaffee man wirklich trinkt... Ansonsten gab es noch einen Kaffee– und einen Snack–Automaten im Pausenraum.

Was bringts?

Ich hatte hier eine Praktikumsstelle, wie ich mir eine Praktikumsstelle vorstelle. Ich konnte Eindrücke der heutigen Arbeitswelt sammeln und mich auch persönlich weiterbilden. Dies war durch Gespräche mit den Mitarbeitern, eine vorzügliche Hardware (Rechner, Handbücher, Räumlichkeiten, Kaffee) und den grossen mir überlassenen Freiraum möglich. Die Teilnahme an einer Gruppensitzung hat mir einen weiteren wertvollen Einblick in die Firma gegeben.

Die Konkretisierung der Begriffe: Qualitätssicherung, Arbeit im Team, Wartbarkeit von Code hat mir persönlich viel gebracht. So habe ich den Programmentwurf aus ganz anderer Sicht gesehen. Während bei Uebungsaufgaben an der ETH nur das Programm laufen muss, ist es in der Praxis wichtig, dass auch Mitarbeiter den Code verstehen und dass Modifikationen ohne grösseren Aufwand möglich sind.

Administratives

Damit das Praktikum anerkannt wird, muss ein Praktikumsbericht erstellt werden. Dieser Bericht muss von der Praktikumsfirma unterschrieben sein und soll enthalten:

- Dauer und Betreuungsperson
- Firmenprofil (Arbeitsumgebung / Atmosphäre)
- Projektbeschrieb
- Persönliche Erfahrungen und Eindrücke
- Würdigung des Resultats

Diesen Bericht müsst Ihr dann im Original bei der Frau Hildgarth abgeben. Zudem muss von der Praktikumsfirma ein spezieller Testatbogen (erhältlich auf dem Abteilungssekretariat) unterschrieben werden.

Diverses

Ich möchte allen Mitarbeitern, mit denen ich direkt zu tun hatte, danken. Namentlich meinem Betreuer Christoph Fleischer, dem Gruppenchef Klaus Ebert und dem Rechenzentrumsspezialisten Rudolf Schütz.

Ich hoffe, dass das NETEA–Team auch weiterhin Praktikanten aufnehmen wird.

Patrick Feisthammel
pfeistha@iiic.ethz.ch

Mattscheibe

(Leserbrief zum Artikel "Bildschirmkommunikation" in den Visionen 4/5)

Als ich vor vier Jahren mein Informatikstudium an der ETH abschloss, war ich der Meinung, dass dieses Klischee des typischen Informatikstudenten (ich brauche bewusst nur die männliche Form) nicht stimmt: Tag und Nacht am Computer, ständig am "Hacken"; sein geistiger Horizont wird links und rechts durch den Bildschirmrand begrenzt. Ich kenne viele ehemalige Studierende der Abteilung IIIC, und auf niemanden trifft diese Beschreibung zu.

Nun werde ich eines Besseren belehrt: Der Artikel von Christian Berger spricht Bände. Wenn das das Niveau des durchschnittlichen Informatikstudierenden im 2. Semester ist – was ich nicht hoffe – dann ist das tragisch für die Abteilung Informatik.

Lieber Christian, wenn Du als Informatikstudent, "der sich oft stundenlang einsam durch Formeln und Codes arbeitet", keine bessere Art der Kommunikation kennst als das "Chatten", und wenn Dein Kontakt zu anderen Menschen im wesentlichen über Multi–User–Adventure–Games erfolgt, dann tust Du mir leid. Du schreibst von Deiner "Beziehung zu den Compis" und merkst offenbar nicht, dass Du auf dem besten Weg dazu bist, die Beziehungen zu Deinen Mitmenschen zu verlieren.

Sicher, man braucht einen Ausgleich zum Studium. Niemand verlangt von Dir, dass Du 24 Stunden täglich Analysisübungen löst und Eltech–Skripts liest. Aber es gibt nun wirklich bessere Freizeitbeschäftigungen als "Gamen","Mudden" und "Chatten" In Zürich gibt es beispielsweise zahlreiche Kinos, Theaters, Konzerte. Es gibt Beizen jeglicher Art, verschiedene Parks und einen See. Falls Deine Finger noch nicht an der Tastatur angewachsen sind, empfehle ich Dir mal einen Ausflug ins Niederdorf oder ans Zürichhorn. Und eine Live–Diskussion mit Kollegen bei einem Glas Bier ist doch ganz was anderes als eine EZINFO–Session über Modem, Tastatur und Bildschirm.

Dass Du in Deinem Adventure–Game bereits "Wizard" bist, freut mich für Dich, aber ich glaube nicht, dass Dir das viel nützt, um auch im Adventure "Vordip 1.0" in die High Scores zu kommen. Ich wünsche Dir jedenfalls viel Glück für Deine Zukunft.

Dani Schnider,
dipl. Informatikingenieur ETH

⇒ Seite 25

Dynamisches Binden

Wenn Ihr Euch die Verzeichnisse anschaut, die in Eurem LD_LIBRARY _PATH angegeben sind, werdet Ihr bemerken, dass oft zwei Files namensgleich sind bis auf die Extensions .a und .so. Nun, auf .a habe ich Euch schon hingewiesen, was soll das .so? Die *.a sind die Bibliotheken für "statisches Linken", die *.so (shared objects) sind dieselben Bibliotheken, allerdings für "dynamisches Linken"[4]. Wie linken wir nun dynamisch? Dazu muss man bei der Link-Phase nur die Option "-dynamic" VOR den dynamisch einzubindenden Bibliotheken angeben. Auf diese Weise kann man statisches und dynamisches Linken auch vermischen: Man kann mit der Angabe von "-static" wieder auf statisches Linken umschalten. Auch diese Option muss vor den statisch zu linkenden Libraries angegeben werden.

Beim Aufruf des Programmes wird automatisch der Betriebssystem-eigene Lader "ld" aufgerufen, der die Bibliotheken hinzulinkt und das Programm startet[5].

[4]"Dynamisches Linken" unterscheidet sich dadurch von "statischem Linken", dass die Libraries erst zur Laufzeit mit dem Programm gelinkt werden. Das spart viel Speicherplatz, dauert aber beim Programmaufruf dafür ein wenig länger. Ausserdem muss man beim Portieren die Libraries mitkopieren.

[5]auch beim statischen Linken benutzt gcc diesen Lader zum Linken.

Make

Nachdem Ihr nun ein wenig mehr über das Linken in Unix Bescheid wisst, denkt Ihr Euch möglicherweise, dass man bei grösseren Projekten leicht den Überblick über die einzelnen Kompilationseinheiten verliert. Dazu kommt, dass man nicht alle Einheiten neu übersetzen möchte wenn man eine Einheit abgeändert hat. Aber eventuell gibt es Abhängigkeiten zwischen den Einheiten - was muss nun neu kompiliert werden? Hier kommt die sagenhafte, beispiellose Lösung: Der Kompilations- und Linkvorgang kann automatisiert werden. Ausschlaggebend ist dabei der Zeitpunkt der letzten Änderung in einem Source-File. Liegt er weniger weit zurück als der Erzeugungszeitpunkt des entsprechenden Objekt-Files, muss neu kompiliert werden. Gelinkt werden muss in jedem Fall (ausser es musste nichts kompiliert werden - dann muss ja überhaupt nichts gemacht werden).

Zur Automatisierung hilft uns "make", das bei jeder Unix-Version dabei ist. Es ist sehr universell und kann auch für andere Programmiersprachen eingesetzt werden. Ich werde hier nur die für normale Kompilationen notwendigen Dinge erklären, alles andere müsst Ihr selbst in der Manpage nachlesen (das Ding kann einfach zu viel).

Make sucht in einer Datei namens "Makefile" nach den Compilationshinweisen. Dieses Makefile hat folgenden Aufbau: Zuerst gibt man eine Regel an, gefolgt von einer Kommandosequenz. Eine Regel hat folgenden Aufbau (in EBNF):

```
Rule ::= Target ":"
Dependencylist.
Dependencylist ::= {Filename "
"}.
Target ::= Filename.
```

In der Dependencylist stehen die Filenamen, von denen das zu erzeugende File abhängt: Wenn wir also ein ausführbares Programm Prog erzeugen wollen, das aus den Objekt-Files Prog1.o, Prog2.o und Prog3. o zusammengelinkt werden muss dann schaut unsere Zeile im Makefile folgendermassen aus:

```
Prog : Prog1.o Prog2.o Prog3.o
       gcc -o Prog Prog1.o
Prog2.o Prog3.o
```

Wie vorher schon erklärt, beginnt in der 2. Zeile die Kommandosequenz. Wichtig ist bei ihr, dass jede Zeile der Kommandosequenz mit einem *Tabulator* beginnt, sonst meldet make einen Fehler. Sie wird ausgeführt wenn das File Prog älter ist, als eines der File, die auf der rechten Seite des ":" stehen. Aber die .o-Files ändern wir ja für gewöhnlich nicht, wir ändern die .c-Files. Also müssen wir beispielsweise für Prog1.o auch zwei Zeilen ins Makefile einfügen (und zwar vor den obigen Zeilen):

```
Prog1.o : Prog1.c
       gcc -c Prog1.c
```

Falls wir nun für jedes Source-File diese zwei Zeilen ins Makefile schreiben müssten, wäre das
relativ happig. Denn viele Sourcefiles müssen ja alle mehr oder weniger auf dieselbe Art compiliert werden. Darum gibt es im Makefile ein paar Abkürzungen:

Die Regel

```
.o : .c
```

ruft immer dann die ihr folgende Kommandosequenz auf, wenn das .o-File älter ist als das entsprechende .c-File. ".c" und ".o" nennt man Suffixe. Es sind mehrere Suffixe vordefiniert. Um herauszufinden, welche anderen es noch gibt, müsst Ihr aber die Manpage konsultieren. Wie sprechen wir unsere Files nun in der Kommandosequenz an? Dafür gibt es folgende Platzhalter im Makefile:

- $* wird durch den Targetfilenamen ohne Suffix ersetzt,
- $@ steht für den vollen Targetfilenamen,
- $< wird durch die gesamte Dependenzliste ersetzt,
- $? vertritt nur diejenigen Files in der Dependenzliste, die jünger sind als das Targetfile.

Wir können also obige Zeilen für alle Sourcecodes verwenden, wenn wir sie im Makefile beispielsweise durch folgende Zeilen ersetzen:

```
.o : .c
     gcc -c $*.c
```

Noch ganz praktisch sind die Makrodefinitionen, die Make unterstützt. Es hat sich als sehr sinnvoll erwiesen, wenn man am Anfang eines Makefiles bestimmte Dinge durch Variablen ersetzt. Z. B. mache ich mir immer folgende Definitionen:

CC= gcc
LN= gcc

In der Kommandozeile schreibt man dann zum Kompilieren nicht mehr gcc, sondern man verwendet beispielsweise

$(CC) -c $*.c

Das hat den Vorteil, dass, sollte man einmal auf einen anderen Compiler umsteigen, man nur die erste Zeile des Makefiles ändern muss. Gleiches gilt auch für den Linker. Mit diesen Makrodefinitionen kann man sich auch am Anfang vom File die Librarielisten und die Objectfilelisten definieren. Gerade bei den Objectfilelisten kann man statt ihrer dann in der Dependenzliste wie auch im Kommandozeilenaufruf das Makro verwenden.

Wenn Ihr das Makefile erzeugt habt, könnt ihr Euer Programm mit

make <Progammname>

kompilieren und linken.

Das Beispielprogramm

Das heutige Beispielprogramm wird uns eine heterogene Liste von Kreisen und Quadraten erzeugen und diese traversieren. Dabei wird uns jedes Element der Liste sagen, was es für ein Datentyp ist und durch welche Parameter es charakterisiert wird. Ausserdem werden alle Elemente ihren Umfang und ihren Flächeninhalt angeben. Ich habe für dieses Programm (auf niedrigem Niveau) objektorientierte Methodik angewandt. Der Supertyp Objekt muss dem Hauptprogramm bekannt sein. Um die abgeleiteten Typen zu definieren, müssen wir den Supertypen kennen und nocheinmal hinschreiben und die zusätzlichen Felder hinzufügen. Von diesen kann das Hauptprogramm nichts wissen. Deshalb greifen wir mithilfe der Prozedurvariablen, die wir im Hauptprogramm definiert haben auf diese Felder zu. Ich weiss, das ist alles ein wenig unsauber, aber anders wäre es zu aufwendig und unnöitg kompliziert.

Das Programm besteht aus drei Kompilationseinheiten: main.c, kreis.c und quadrat.c. Ausserdem habe ich noch ein Makefile angegeben, das dieses Programm kompiliert. Zum Laufen bringt ihr das Ding mit :

```
make objects
objects
```

Das war die vorläufig letzte Folge des C-Kurses. Wir meinen, dass wir Euch jetzt alles, was Ihr über C wissen müsst, beigebracht haben. Falls Ihr da anderer Meinung seid und meint, wir müssen unbedingt noch das eine oder andere Thema besprechen, dann meldet Euch bei uns (mail an leonhard@vis.inf.ethz.ch). Tja, das wärs und: Ob wir an diesen Kurs noch einen C++-Kurs dranhängen, hängt davon ab, wie schnell wir es lernen. So long

lj

```
/*************************************************/
/* Datei: main.c                                 */
/*                                               */
/* Hauptmodul des Beispielprogrammes "Objects"   */
/* zur 4. Folge                                  */
/* des visionaeren C-Kurses.                     */
/*                                               */
/* Autor: Leonhard Jaschke                       */
/*************************************************/

#include "Object.h"

#define N 3

extern void KreisNeu(Object **kopf, float x, float y,
float a);
extern void QuadratNeu(Object **kopf, const char[], float
r);

void Traversier(Object *kopf)
{
      Object *t;
      float u,f;
      char s[80];
      int i;

      i= 0;
      t= kopf;
      while(t!=NULL)
      {
          t->ident(t,s);
          u= t->umfang(t);
          f= t->flaeche(t);

          printf("%2d\t%s\n",i,s);
          printf("\tUmfang: %f\t Flaeche: %f\t\n\n",u,f);
          t=t->next;
          i++;
      }
}

void main(int argc, char *argv[])
{
      Object *kopf= NULL, *t;
          /* Kopf der Liste, sie wird als Stack */
          /* implementiert (ist am einfachsten) */
```

```
        /* t ist ein Hilfselement */
    int i;
    float x,y,r; /* Koord.von Kreis, Radius des */
                 /* Kreises */
    float a;  /* Seitenlaenge des Quadrats */
    char name[30]="Region ",append[3],realname[30];
                 /* Name des Quadrats */
    /* Wir erzeugen abwechselnd einen Kreis und */
    /* ein Quadrat */
    for (i= 0; i<N; i++)
    {
        x= (float)i*3.0+2.0;
        y= SQR((float)i+1.0);
        r= 10.0-(float)i;
        a= r/2.0;
            /* irgendwelche zufaelligen Formeln */

        sprintf(append,"%d",i);
        strcpy(realname,name);
        strcat(realname,append);

        KreisNeu(&kopf,x,y,r);
        QuadratNeu(&kopf,realname,a);
    }

    Traversier(kopf);

    while(kopf!=NULL)
    {
        t= kopf;
        kopf= kopf->next;
        free(t);
    }

}
```

```
/*************************************************/
/* Datei: kreis.c                                */
/*                                               */
/* Modul fuer Kreise des Beispielprogrammes      */
/* "Objects" zur                                 */
/* 4. Folge des visionaeren C-Kurses.            */
/* Autor: Leonhard Jaschke                       */
/*************************************************/

#include "Object.h"

#define PI 3.14159265

typedef struct NeuOb
{
      void (*ident)(Object *me, char s[]);
      float (*umfang)(Object *me);
      float (*flaeche)(Object *me);
      struct Supertype *next;
      float x,y,r;
} Kreis;

static void Ident(Object *me, char s[])
{
      Kreis *k;

      k= (Kreis *)me;
      sprintf(s,"Kreis, x: %f, y: %f, r: %f",k->x,k->y,k-
>r);
}

static float Umfang(Object *me)
{
      Kreis *k;

      k= (Kreis *)me;
      return(2*k->r*PI);
}

static float Flaeche(Object *me)
{
      Kreis *k;

      k= (Kreis *)me;
      return(SQR(k->r)*PI);
}
```

```
void KreisNeu(Object **kopf, float x, float y, float r)
{
      Kreis *new;

      new= (Kreis *)malloc(sizeof(Kreis));
      new->x= x;
      new->y= y;
      new->r= r;
      new->ident= Ident;
      new->umfang= Umfang;
      new->flaeche= Flaeche;
      new->next= *kopf;
      *kopf= (Object *)new;
}



/*******************************************************/
/* Datei: quadrat.c                                    */
/*                                                     */
/* Modul fuer Quadrate des Beispielprogrammes "Objects"*/
/* zur 4. Folge des visionaeren C-Kurses.              */
/* Autor: Leonhard Jaschke                             */
/*******************************************************/
#include "Object.h"

typedef struct NeuOb
{
      void (*ident)(Object *me, char s[]);
      float (*umfang)(Object *me);
      float (*flaeche)(Object *me);
      struct Supertype *next;
-->   char name[30];
-->   float a;
} Quadrat;

static void Ident(Object *me, char s[])
{
      Quadrat *q;

      q= (Quadrat *)me;
-->   sprintf(s,"Quadrat, name: %s, a: %f",q->name,q->a);
}

static float Umfang(Object *me)
{
```

```
        Quadrat *q;

        q= (Quadrat *)me;
        return(4*q->a);
}

static float Flaeche(Object *me)
{
        Quadrat *q;

        q= (Quadrat *)me;
        return(SQR(q->a));
}

void QuadratNeu(Object **kopf, const char name[], float
a)
{
        Quadrat *new;

        new= (Quadrat *)malloc(sizeof(Quadrat));
        strcpy(new->name, name);
        new->a= a;
        new->ident= Ident;
        new->umfang= Umfang;
        new->flaeche= Flaeche;
        new->next= *kopf;
        *kopf= (Object *)new;
}


/*******************************************************/
/* Datei: Object.h                                     */
/*                                                     */
/* Headerfile zum Beispielprogramm "Objects" zur       */
/* 4. Folge des visionaeren C-Kurses.                  */
/* Autor: Leonhard Jaschke                             */
/*******************************************************/

#define SQR(X)      ((X)*(X))
#define NULL  (0L)
/* Dass 'L' steht fuer long und kennzeichnet die 0 als
long-Wert   */

typedef struct Supertype
{
/* Hier werden die Prozedurvariablen definiert         */
```

```
/* Leider ist es nicht moeglich (wie bei meinem        */
/* Universalrezept),  Prozedurtypen zu definieren, weil*/
/* wir rekursive Abhaengigkeiten zwischen dem Supertyp */
/* und den Prozedurtypen haben. So universal war das   */
/* Rezept also doch nicht *Sniff*                      */
      void (*ident)(struct Supertype *me, char s[]);
      float (*umfang)(struct Supertype *me);
      float (*flaeche)(struct Supertype *me);
/* und hier folgt noch eine "normale" Variable: */
      struct Supertype *next;
} Object;
```

```
#########################################################
# Datei: makefile                                       #
#                                                       #
# Makfile des Beispiel Programmes "Objects" zur 4. Folge#
# des visionaeren C-Kurses.                             #
#                                                       #
# Autor: Leonhard Jaschke                               #
#########################################################

CC=cc
LN=cc

quadrat.o: quadrat.c Object.h
      $(CC) -c quadrat.c

kreis.o: kreis.c Object.h
      $(CC) -c kreis.c

main.o: main.c Object.h
      $(CC) -c main.c

objects: main.o kreis.o quadrat.o
      $(LN) -o objects main.o kreis.o quadrat.o
```

Aus der Abteilung für Informatik, IIIC

Wir möchten es nicht unterlassen, Ihnen die Bedingungen zur Kernfachprüfung Digitaltechnik und Rechnerstruktur im Herbst 1994 auch auf diesem Wege nochmals bekanntzugeben:

Kernfach-Prüfung Digitaltechnik und Rechnerstruktur im Herbst 1994

für die Prüfung in Digitaltechnik und Rechnerstruktur (Kernfach Hardware) im Herbst 1994 gelten folgende Regeln:

1. Studenten, welche nach dem neuen Studienplan abschliessen, werden von Professor Eberle geprüft. Die Prüfung setzt sich aus einer Schnellübung und der regulären Prüfung zusammen. Die Schnellübung wird zu einem Drittel, die reguläre Prüfung zu zwei Dritteln in der Endbewertung berücksichtigt. Zur regulären Prüfung zugelassen wird nur, wer die Schnellübung abgelegt hat. Wer nicht an der Schnellübung teilnehmen kann (WK, Krankheit, etc.) wird zu einem späteren Termin während des Semesters nachgeprüft.

2. Studenten, welche sich bei Professor Eberle nach dem alten Studienplan prüfen lassen, haben keine Schnellübung abzulegen und treten lediglich zur regulären Prüfung an, welche zu 100% die Endbewertung bestimmt.

3. Studenten nach Studienplan 1989, welche ein Testat von Herrn Dr. Zinniker erhalten haben, können sich bei Herrn Dr. Zinniker prüfen lassen. Wie bis anhin werden diese Kandidaten zusammen mit den El'Ing.-Studenten von Prof. Fichtners Leuten geprüft. Falls gewünscht, können sich Studenten mit einem Testat von Herrn Dr. Zinniker auch bei Prof. Eberle prüfen lassen. Von solchen Kandidaten muss wiederum nur die reguläre Prüfung abgelegt werden, welche zu 100% die Endbewertung bestimmt.

Hilfsmittel zur Prüfung von Professor H. Eberle:
alle Hilfsmittel erlaubt
Hilfsmittel zur Prüfung von Dr. R. Zinniker:
sämtliche schriftlichen Unterlagen/Taschenrechner

Bitte schon vormerken:
Testat- und Zulassungskontrolle für die Prüfungssession Herbst 1994
Die Testatkontrolle (Studierende nach Studienplan 1989) sowie die Zulassungskontrolle (Studierende nach Studienplan 1993)

findet in der letzten Semesterwoche
von Montag, 11. Juli, bis Donnerstag, 14. Juli, statt.

Obschon Studierende nach neuem Studienplan 1993 keine Testate mehr erhalten, haben sie sich zur Zulassungskontrolle unbedingt auch zu melden. Die Zulassungskontrolle dient vor allem der Prüfung, ob die angemeldeten Fächer mit denjenigen von der Prüfungsplanstelle gemeldeten identisch sind. Allfällige Differenzen können so noch rechtzeitig bereinigt werden; auch können Teilabmeldungen von einzelnen Fächern zu diesem Zeitpunkt noch beim Abteilungssekretariat erfolgen.

Mentorsystem
Den neu ins Fachstudium Eintretenden möchten wir das Mentorensystem in Erinnerung rufen. Die Studierenden sollen ihre persönlichen Studienpläne regelmässig mit einem selbstgewählten Mentor besprechen. Dies dient unter anderem der Förderung der persönlichen Kontakte zwischen Professoren und Studierenden. Soweit möglich, sollten Sie einen Professor aus jener Fachrichtung wählen, in der Sie sich vertiefen wollen. Die Listen zur Anmeldung liegen auf dem Abteilungssekretariat auf. Dieses vereinbart dann auch die Termine mit dem Mentor.
Bereits mit einem Mentor in Verbindung stehenden Studierenden empfehlen wir, mit fortschreitendem Studium die Beratung kontinuierlich zu beanspruchen.
Gleichzeitig möchten wir darauf hinweisen, dass für die Zusammenstellung des Nebenfachs der Fachberater und nicht der Mentor zuständig ist. Alle Vorbesprechungen dazu laufen über den Abteilungssekretär, Herrn Helmut Kaufmann, der dann die Anträge dem Fachberater zur Genehmigung unterbreitet.

Das Abteilungssekretariat

Nachfolgerin oder Nachfolger für das Ressort "Prüfungen & Unterricht" gesucht

Meine Zeit an dieser illustren Abteilung geht (beinahe) dem Ende zu: Ich muss nächstes Semester (hoffentlich!) nur noch die Diplomarbeit schreiben. Tja, und so eine Diplomarbeit braucht Zeit, was sich schlecht auf die Betreuung meines Ressorts auswirken würde. Also habe ich mir selbst den Rücktritt gegen Ende dieses Semesters nahegelegt. Die Lücke, die ich hinterlassen werde, gilt es zu füllen: Vielleicht hast Du ja Lust, mein Ressort zu übernehmen?

Wichtig ist für dieses Ressort, dass man Spass hat anderen Studierenden in ihrem Studium zu helfen, und die Tatsache, dass ich mit meiner Arbeit im VIS anderen Studierenden geholfen habe, hat mich für den manchmal grossen Aufwand genug entschädigt. Was hat man sonst noch davon, dass man im VIS-Vorstand tätig ist? Man lernt viele Leute kennen, man lernt Sachen zu organisieren, man hat Kontakt zu den Professoren, man hat die Möglichkeit etwas zu bewirken, man erfährt, was in der Abteilung und im Departement los ist, man darf gratis auf die VIS-Feste und zum VIS-Mitarbeiteressen.

Die wichtigste Arbeit in meinem Ressort ist es, die Vordiplomprüfungen und Schlussdiplomprüfungen zu sammeln, in den Ordnern zu archivieren und die Vordiplome als Handouts herauszugeben. Andere Aktivitäten, wie beispielsweise die Vorlesungsumfragen, die Prüfungsstatistiken, die Organisation von Lerngruppen, die Beschaffung und Verwaltung von Skripten fallen zwar auch in dieses Ressort, aber vieles davon ist sehr zeitintensiv, so dass diese Aktivitäten oft von anderen VS-Mitgliedern (zumindest zum Teil) übernommen worden sind.

Im Grunde wird ein Platz im Vorstand frei, das heisst also nicht unbedingt, dass mein Ressort übernommen werden muss. Wenn Du Lust hast an einer anderen Tätigkeit im Vorstand, kann es ohne weiteres sein, dass im Vorstand die Ressorts getauscht werden und jemand anderer mein Ressort übernimmt. Also, selbst wenn Du keine Lust hast mein Ressort (aber dafür ein anderes) zu übernehmen, dann schau doch mal bei uns vorbei.

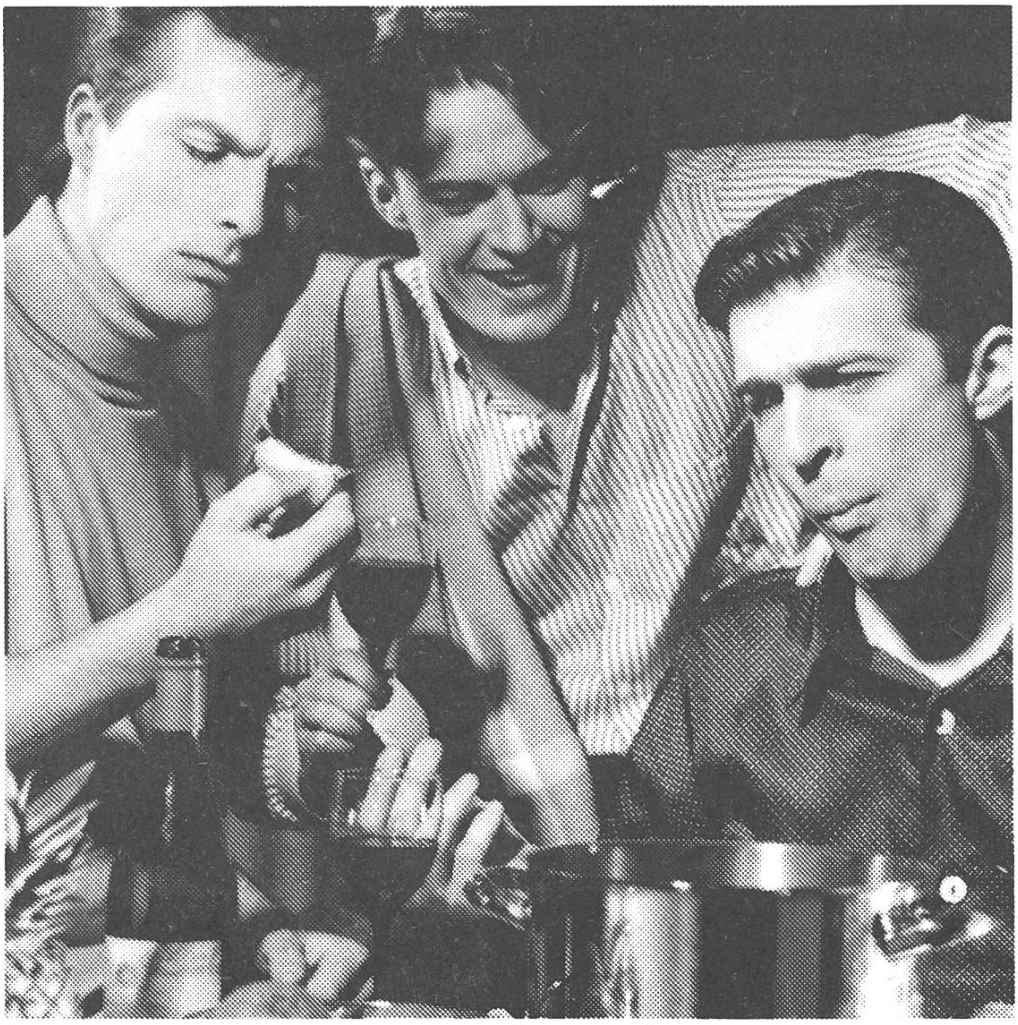
Ich möchte all denjenigen danken, die mich in meiner Arbeit im VIS-Vorstand unterstützt haben mit dem erstellen von Musterlösungen, Skripten, mit dem Organisieren der Lerngruppen. Ganz herzlicher Dank gilt natürlich den Leuten im VIS-Vorstand, die meine Launen ausgehalten und mich kräftig in jeder Hinsicht unterstützt haben.

lj

UBS PROFESSIONAL

Einstieg für Informatik-Ingenieure bei der SBG.

Auch mein Projektleiter hat den Menü-Vorschlag ganz exquisit gefunden.



Wenden Sie Ihre Informatik-Kenntnisse beim grössten Schweizer Software-Spezialisten an. Ein spezielles Ausbildungsprogramm macht Sie mit der vielfältigen Hard- und Software bekannt. Es zeigt Ihnen auch, wie breitgefächert Ihre Entwicklungschancen bei der SBG sind. Rufen Sie einfach Herrn Jürg Winzenried an. Er wird Sie on line über Karrieremöglichkeiten informieren.

☏ **SBG-JobInfo, Herr Winzenried: 01/236 70 45**

Wir machen mit.

SBG UBS Schweizerische Bankgesellschaft

Chris Flu's Kochecke

13. Folge: Teigwaren mit Zucchettirahmsauce oder: wehret den Anfängen

Meine Tage sind gezählt. Einst hoffte ich, nach Abschluss des Studiums die prallen Früchte meines Schaffens ernten zu können. Nichts mehr sollte mich auf meinem Weg zu den Sternen aufhalten, Zügel frei, volle Fahrt voraus ins Glück. Ich will es gleich vorwegnehmen: im Moment versuche ich eine der letzten Hürden, welche sich einem/einer InformatikstudentIn ins Ethernet stellt, zu nehmen – ich mache meine Diplomarbeit. Nun ist dies keineswegs ein Grund zur Sorge, hätte nicht eines schönen Tages irgendeinE InformatikerIn (vielleicht sogar so ein, kalt fliesst mir der Schweiss den ungewaschenen Rücken hinunter, Hacker (Schande über mein Haupt, dass ich es gewagt habe auch nur daran zu denken)) ein Tool entwickelt, welches sich fortan Mail nannte. Niemals hätte ich geglaubt, dass ich den Gebrauch dieses Kommunikationsmittels dereinst mit solch hohem Einsatz zu bezahlen haben würde.

Es begann alles ganz harmlos (wie immer im Leben). Pleoni und ich tauschten von Zeit zu Zeit die wichtigsten Resultate unserer Forschungstätigkeit (wie mache ich, dass im X–Windows der Iconmanager auf dem Bildschirm oben links statt unten rechts erscheint) aus und zwar, weil wir notgedrungenermassen den Tag an einer Sun SPARCclassic statt an der Sonne (uaaahhhh, der Scherz ist ja rieeesig) verbringen, per E–Mail, wie sich das für eineN Infostudi im Diplomsemester gehört.
Etliche von Euch werden das auch schon getan und als absolut ungefährlich klassiert haben. Dem ist nicht so! Wehe mir, der ich die Dämonen rief. Vor zwei Wochen hatte ich die zugegebenermassen nicht sonderlich intelligente Idee, mir selbst eine Mail zu senden, um die Performance des Netzes zu testen. Das ging ohne grössere Probleme vonstatten, der einzige knifflige Punkt war die Frage, wie ich mir etwas schreiben sollte, das ich nicht bereits wusste. Abgeschickt, gewartet, es funktionierte: wenig später traf die Mail wieder bei mir ein. Nun denn, wenn das so einfach ist, dachte ich mir, könnte ich die Mail mit Reply gleich selbst beantworten. Jetzt kommt's. Die Antwort kam niiiiieeeee an! Dazu muss man/frau wissen, dass bei meiner Replyadresse irgendetwas nicht stimmt, sie wird beim Absenden jeweils gekappt, es kann mich also niemand mit der 'Reply–Funktion' seines Mail–Tools erreichen. Was zurückkommt ist eine Meldung folgender Art:

```
Return-Path: MAILER-DAEMON
Received: by mint.inf.ethz.ch
    (5.65c/ETHZ-3.1) id
AA14697; Mon, 9 May 1994
22:03:17 +0200
Date: Mon, 9 May 1994 22:03:17
+0200
From: Mail Delivery Subsystem
<MAILER-DAEMON>
Message-Id:
<199405090703.AA14697@mint.inf
.ethz.ch>
To: flueler
Subject: Returned mail: User
unknown

   ----- Transcript of session
follows -----
While talking to
neptune.ethz.ch:
>>> RCPT
To:<testuser@inf.ethz.ch>
<<< 550
<testuser@inf.ethz.ch>... User
unknown
550 <testuser@inf.ethz.ch>...
User unknown
```

(...)

Nun beginnt ein logisch denkender Mensch aber etwas zu wittern: erstens steht da klar und deutlich "MAILER-DAEMON" und erst noch grossgeschrieben. Zweitens: wie soll dieser Mailer-Daemon meine Reply an mich zurücksenden können mit dem Vermerk, der User sei unbekannt, wenn auch der Absender der Mail (dank Verstümmelung der Replyadresse) unbekannt ist? Drittens: was will uns das sagen? Was ich schon immer vermutet habe, scheint sich nun zu bestätigen. Es sind weder Elektronen noch Spannungsdifferenzen, die alle unsere Maschinen zum Laufen bringen, es sind DAEMONEN! Wiederwärtige kleine Biester, die in windeseile Windows auf die Bildschirme zaubern und Berechnungen durchführen. Niemand wage sich in ihr Reich vor. Es ergeht ihm gar schrecklich. So höret denn das Ende meiner Geschichte. Gehen wir nochmals einen Schritt zurück: besagte Mail erreichte ihr Ziel (mich!) nie. Sie wird seither mit wachsender Geschwindigkeit zwischen zwei Dämonen hin- und hergeschoben, wobei jeder dem anderen klar machen will, dass seine Mailadresse nicht stimmt. Dies manifestierte sich zuallererst in einem deutlichen Performance - Verlust meiner Maschine, was mich schon mal stutzig machte. Kurz darauf stürzte bei uns im UBILAB (wo ich meine Diplomarbeit schreibe) der Server ab. Ich fühlte mich der Sache nicht mehr gewachsen und schrieb folgende Mail an Pleoni (seines Zeichens Ex-Redaktor der Visionen):

```
To: Patrick Francesco Leoni
<pleoni@iiic.ethz.ch>
Subject: Daemon
```

---

hm, so kleine bemerkung, soeben ist bei uns tatsaechlich der server abgestuerzt und prof. marti ist am rande der verzweiflung, weil ihm dabei eine ganze MAIL geloescht worden ist, bevor er sie abschicken konnte. wie war jetzt das mit dem maildaemon; die geschichte beginnt sich langsam

zu verselbstaendigen, vielleicht muessten wir da mal ein journalistisches auge drauf werfen...
chris flu

p.s. mir ist die sache nicht mehr ganz geheuer, vielleicht sollte ich mich mal versetzen lassen...

Seine Antwort:
Im Ernst, ich steck nicht dahinter, das haben die diversen Daemonen auf ihrer woechentlichen Sitzung selbst entschieden. Sie werden jetzt, so als Reaktion auf Dein Falsche –Adresse –Experiment von vorletzter Woche, immer wieder abstuerzeln (oder manchmal nur etwas straucheln...); als naechstes ist der telnet–daemon dran, der schickt so als Spass alle Zeichen, die Du tippst, erst einmal zum Bruder – Daemon in zmitzt.im. outback.au, und der schickt sie dann zurueck, was in einer grausamen Verzoegerung resultiert; nachher kommt der maus–daemon dran (so in etwa am Freitag), dann bewegt sich der Mauspointer ploetzlich von allein, und die Maus huepft so auf und ab, spaeter ist der Tastaturdaemon an der Reihe, der zieht dann mitten, waehrend Du tippst, einzelne Tasten hinunter, und schreibt Dir so ICH eiBINn paDER TASTar MelATURDAEdungen aMONuf Deinen Bildschirm, mitten in die Woerter, die Du schreibst, ganz schoen gruselig, wenn ploetzlich die Tasten sich von alleigen druecken; dann kommt der Diskdaemon dran, der schreddert dann die eingelegten Disketten schoen saeuberlich bevor er sie wieder auswirft (gemein ist, dass er das erst tut, wenn Du "eject" eingegeben hast...); dann spinnt der Save–Daemon, und stuerzt ab, gerade kurz bevor er Deine 25 Kilo DA–Bericht erstmals aus dem Editor abspeichern soll; und zu guter letzt spielt der Diskdriverdaemon mit der Hardddisk Frisbee im Computer-gehaeuse, was der Maschine den Garaus macht. Aber jetzt keine Angst, das alles ist nicht so schlimm, wenn Du rechtzeitig Abbitte tust, so ueberlegen sich's die Daemonen noch einmal, nicht wahr, die lassen mit sich reden, sind ja auch nur Daemonen. Also schick doch eine Meldung an flueler@mint.inf.ethz.ch mit dem Inhalt "Ich schreibe keine Meldung an flueler@mint.inf.ethz.ch" (cklewer, hae!), und, essentiell, wenn sie ankommt, beantworte sie NICHT. Dann hast Du den maildaemon besaenftigt, und der reportiert das dann den anderen Daemonen, und die sind wieder friedlich und marti nicht mehr verzweifelt.
Alles clear? Then wish I you a good afternoon Patrick

Um es gleich vorwegzunehmen: es klappte nicht. Ich ging sogar noch weiter als Pleoni vorgeschlagen hatte und sandte folgende Mail:

```
To:flueler@mint.inf.ethz.ch
Subject:mea culpa
```

---

seid unterthaenigst gegruesset, oh ihr goetter, die ich rief. die schuld, die ich

auf mich geladen, nicht wissend, welch ungemach dabei euch widerfahren wuerde, lastet tonnenschwer auf meinem schwachen ruecken, wie kann ich, ach, fuer mein vergehen abbitte leisten, euch zu besaenftigen, herrscher ueber himmel und netz. vergebung sei mir gewaehret, mein zaghaft herz ruft euch klamm entgegen, habet nachsicht mit einem unbedeutenden anwender, der sich einen schritt zu weit vorgewagt hat und nun ueber unendlichem abgrund waehnet. so pruefet denn mein gesuch und stellet eure forderungen, ich will sie euch erfuellen, so weit dies meiner klaeglichen rasse moeglich ist, denn ach wie viel maechtiger seid ihr, die mail–daemonen.
hochachtungsvoll chris flu

Sie blieb ohne Wirkung, im Gegenteil: kurz darauf begannen sich erste S&onderze#ichen ins Sch@riftb*ild einz&umischen, bald gSAUSanzTUe SäDItze, wobei über das Niveau der UrheberInnen kein Zweifel bestand. Eine letzte Mail konnte ich an Pleoni senden, in der ich ihm die Tragweite meines Vergehens schilderte:

(...) irgendwann wird das netz ueberlastet sein, zuerst wird der datentransfer innerhalb der eth zusammenbrechen, saemtliche grossrechner–anwendungen werden abstuerzen, kalkulationen, die seit tagen, vielleicht gar jahren in berechnung sind, muessen nochmals von vorne gestartet werden. das ganze schweizerische telekommunikationsnetz wird überlastet, mehrere verteilzentralen werden durchbrennen, die kommandostände der kernkraftwerke werden von den reaktoren getrennt, diese ueberhitzen sich, explodieren, die schweiz wird ein radioaktives truemmerfeld... und alles wegen mir (schluchz, ich werde mir nie mehr eine mail schreiben, ich versprech's hoch und heilig bei djikstra und steve jobs und adolf ogi!).

Die Leitung ist nun tot, einzelne Dämonen begannen aus dem Diskettenlaufwerk zu kriechen, ich unternahm letzte Schlichtungsversuche; sie blieben erfolglos. Merkwürdigerweise überraschte es mich nicht einmal mehr, als sie mich zu sich ins Gehäuse zogen. Sie sind gut zu mir. Es ist ein wenig eng. Wenn sie schlafen, versuche ich manchmal wenigstens ans Netz zu kommen. Hoffentlich wird diese Mail ihr Ziel erreichen. Sendet bitte auch liebe Grüsse an meine Eltern, es geht mir gut, auch an den Pelz und die langen Ohren, die mir langsam wachsen, habe ich mich gewöhnt. Bitte starrt nicht so in den Bildschirm, wenn Ihr hackt, ich fühle mich jeweils so beobachtet und frage mich, ob das Window wohl schief aufgehängt ist oder ob ich zu langsam war, aber alleine ist man manchmal schon ziemlich im Stress.

Aus Tradition folgt nun ein Rezept, welches sich vieltausendfach bewährt hat, wenngleich es für mich schwierig

ist abzuschätzen, ob es schmeckt oder nicht. Im Moment mag ich vor allem Kupfer. Auch Magnetband ist nicht zu verachten, aber da kommt man nicht so gut ran. Ganz vorzüglich schmecken uebrigens die Bugs in den neuen Intel–Chips, da kann man sich auch ziemlich satt essen von.

Teigwaren mit Zucchettirahmsauce

Man/frau benötigt (für 1–2 StudentInnen):

2 Zucchetti
1 Knoblauchzehe
1/2 – 1 Bouillonwürfel
1 dl Rahm

Wie Spaghetti (oder Equivalens) gekocht werden müssen, hoffe ich nicht zu erklären genötigt zu sein. Für Eingefleischte folgt ein sicherer Algorithmus ohne nähere Erläuterungen:

```
PROCEDURE Spaghetti(VAR Salz,
Wasser, Spaghetti: Zutaten);

VAR NOSIP: INTEGER;
   Result: CookingState;

BEGIN

Result := 'hard & heavy';
SYSTEM.PUT(Pfanne,Wasser);
SYSTEM.SET(KochplatteWherePfann
eIsStandingOn.Heat);
REPEAT UNTIL Wasser = Boiling;
SYSTEM.PUT(Pfanne,Salz);
SYSTEM.PUT(Pfanne,Spaghetti);
WHILE (Result <> 'al dente') AND
(NOSIP <> 0) DO
Probieren(Spaghetti,Result);
NumberOfSpaghettiInPfanne(NOSIP)
;
END; /* WHILE */
SYSTEM.CLEAR(KochplatteWherePfa
nneIsStandingOn.Heat);
Separate(Spaghetti,Wasser);
END Spaghetti;
```

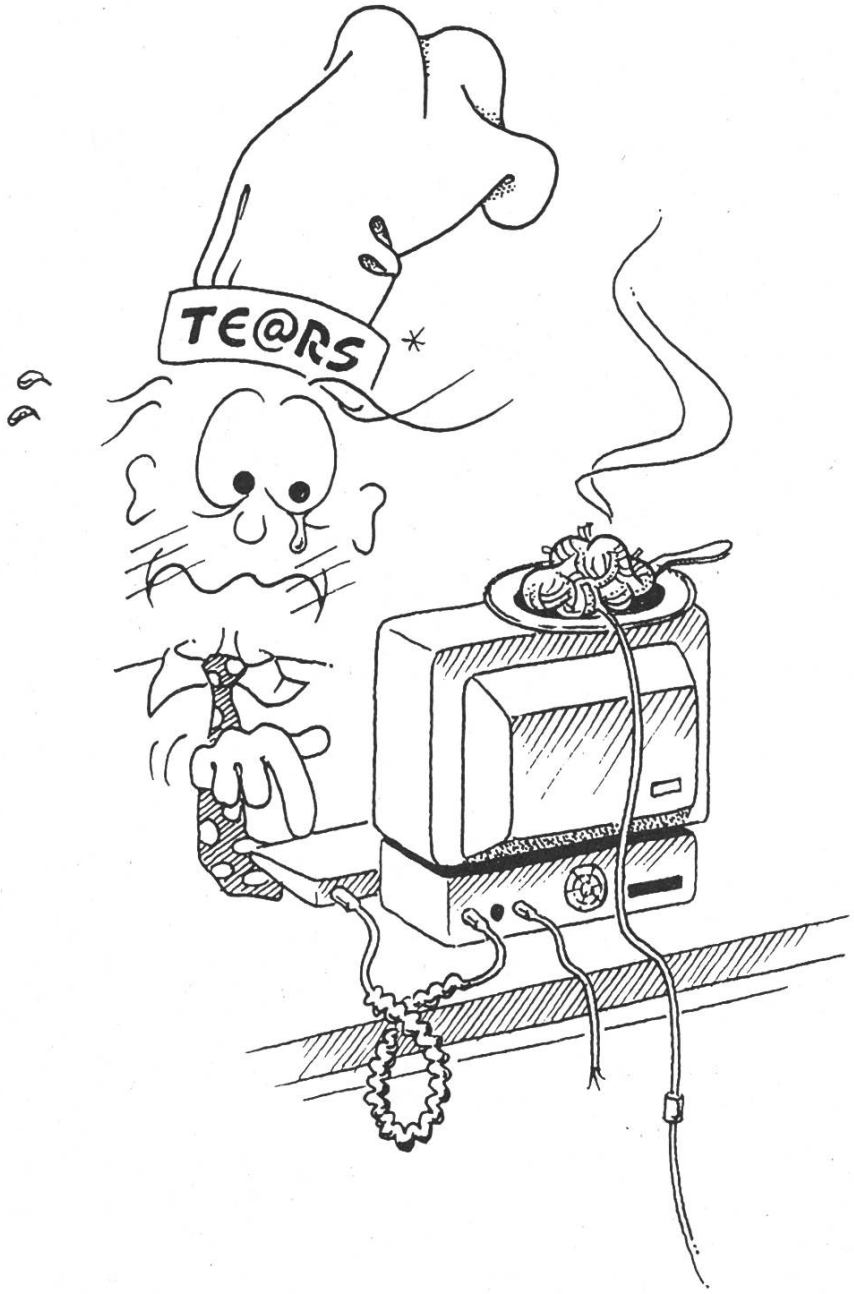
Die genaue Ausformulierung der Prozeduren

```
Probieren(Input: Zutaten; VAR
Result: Essbar);

NumberOfSpaghettiInPfanne(VAR
NOSIP: INTEGER);

Separate(VAR Itme1,Item2: Zutaten);
```

sei wie üblich der geneigten Leserschaft überlassen. Man/frau mache sich auch Gedanken über die genaue Definition der Konstanten 'Boiling' und 'Pfanne' sowie zur Datenstruktur von Zutaten, KochplatteWherePfanneIsStandingOn und CookingState. Ist es sinnvoll, dass KochplatteWherePfanneIsStandingOn global definiert wird? Es soll weiter geprueft werden, ob CookingState nicht auch als zweiwertige Menge (Boolean) mit den Elementen 'cooked' = TRUE und 'uncooked' = FALSE ausgeführt werden koennte. Welches sind Vor– und Nachteile einer solchen Abstraktion?



* Manchmal ist es echt zum Weinen:
Christin schickte mir seinen Text per E-Mail
– eigentlich eine gute Idee. Was jedoch an-
gekommen ist, g%eicht @inem ~ mi&&leren
Ch@os (~eh%@rh!f+ , u&-@&!+ ... (@.!+ ~

M@sus.
1994

Die Sauce ist wesentlich einfacher zuzubereiten: Schneide die Zucchetti in Würfel, gib sie mit dem Bouillonwürfel und dem (gepressten) Knoblauch in eine Pfanne, koche die Mischung so lange, bis die Zucchetti leicht mantschig sind (10–15 Minuten), puriere sie mit dem Stabmixer (wenn Du keinen hast, kannst Du auch versuchen, sie mit der Gabel zu zerdrücken; ist aber nicht zu empfehlen (ich habe einen Mixer (har, har)), gib den Rahm dazu, wärme die Sauce nochmals kurz auf (nicht kochen), fertig.
Tip: Statt Zucchetti können auch Auberginen, Aepfel oder Hardware – Skripte genommen werden. Ein paar Tropfen Zitronensaft geben der Sauce noch mehr Rasse. Zum Würzen empfehle ich Basilikum oder Salbei.

Ich wünsche Euch viel Freude an diesen irdischen Genüssen – und denkt an mich, wenn Ihr über die lahme Maschine flucht. Es ist manchmal wirklich anstrengend und der Lohn ist lausig...

Chris Flu

Ihr habt sicher bemerkt, dass seit der letzten Kochecke relativ viel Zeit verstrichen ist. Wie wir erfahren mussten, ist Chris Flu eine Woche vor Redaktionsschluss spurlos verschwunden. Die vorliegende Kochecke scheint das letzte Lebenszeichen, welches wir von ihm erhalten haben, zu sein. Ein Verbrechen kann nicht ausgeschlossen werden. Meldet uns bitte, wenn Euch irgendwelche besonderen Vorkommnisse aufgefallen sind (die Red.).

Viskas? Na klar, Viskas!

fm. Um es vorwegzunehmen: mit Katzen-futter hat das Viskas wenig zu tun, aber mit Informatikerfutter! Denn das Viskas ist ja schliesslich DAS Ereignis am Katzensee, an dem es Spanferkel, Kotelettes, Würste, Bier, und gute Laune gibt. Das alles organisiert natürlich der VIS exklusiv für Dich und Deine Freunde.
Und ich habe gehört, dass das Viskas dieses Jahr auch was ganz besonderes werden soll, denn der VIS ist ja 10 Jahre alt geworden. Mehr darf ich aber noch nicht verraten.
Also, reservier Dir jetzt sofort den Termin der Very Important Session at KAtzenSee! Gleich. Für immer. Unwiderruflich.
Diese Einladung gilt natürlich wie immer auch für alle Assistentinnen und Assistenten, Dozentinnen und Dozenten unserer Abteilung.

Wir freuen uns jetzt schon auf Euch und die Spanferkel!

WANN ?
WO ?
am 16. Juni 1994
ab Nachmittag
Bei Sonne: am Katzensee
Bei Regen: im Katzensee
Bei Schnee oder Hagel: am Katzensee
Bei Tornado oder Erdbeben: am Katzensee
Bei Weltuntergang am 16. Juni:
am 15. Juni am Katzensee
VSAS

G.A.B. 6648 Minusio

Falls unzustellbar bitte zurück an:

Verein der Informatikstudierenden
IFW B29
ETH–Zentrum

CH–8092 Zürich

Inhalt

Adressen	S. 2
Hei Folkens!	S. 3
Ministerwechsel in F&K	S. 5
FTP, Archie, Gopher & WWW	S. 6
Prüfungszusammenfassung	S. 11
C-Kurs, die Vierte(I)	S. 22
Praktikum bei der ABB Netcom AG	S. 27
Leserbrief	S. 30
C-Kurs, die Vierte (II)	S. 31
Abteilungsnews	S. 40
Gesucht: NachfolgerIn	S. 42
Chris Flu's Zucchettisauce	S. 44
VISKAS	S. 50