

inforum

INFORmationsforum des Rechenzentrums der Universität Münster

Jahrgang 11, Nr. 3 – Juli 1987

ISSN 0931-4008

Inhalt

Editorial	2
<u>RUM-Aktuell</u>	
Netz-Memorandum der DFG	3
QN – ein Ersatz für QUERY NAMES	4
Beschränkungen auf der C-Maschine	5
Das USERNAME-Kommando	5
Reduzierung der RSCS-Meldungen	5
Skripten	6
Zusätzliche Steuerkarten bei SPSS ^X	6
Personalia	8
REDUCE im MVS verfügbar	8
NAG-Bibliothek im UTS	9
Ein neuer GML-Befehl – :language	9
<u>RUM-Lehre</u>	
Lehrveranstaltungen im 2. Halbjahr 1987	11
<u>RUM-Graphik</u>	
Präsentationsgraphik in FORTRAN	18
GVT – ein neues Terminalprogramm für IBM-PCs	21
<u>RUM-Tutorial</u>	
Bildschirmsteuerung im Dialog mit ISPF	23
<u>RUM-Aroma</u>	
inforum -Quiz	26
Känzeln Sie Ihren Auswurf?	26

Impressum**inforum**

ISSN 0931-4008

Redaktion: A. Achilles (Tel. 83-2607)
 W. Bosse (Tel. 83-2461)
 H. Pudlatz (Tel. 83-2472)
 E. Sturm (Tel. 83-2609)

Satz: E. Krause
 L. v. Wüllen

Druck: H. Mecke

Universitätsrechenzentrum
 Einsteinstraße 60
 4400 Münster

Auflage dieser Ausgabe: 1000

Redaktionsschluß der nächsten
 Ausgabe: 30.9.1987

Editorial

von

H. Pudlatz

Diese Ausgabe hat Sie hoffentlich noch vor der Fahrt in die Sommerferien erreicht, damit Sie sie noch vor Beginn des Wintersemesters lesen und so – mit allem Wissenwerten über unser Lehrangebot und allen aktuellen RUM-Neuerungen bekanntgemacht – den rechten Gebrauch von den RUM-Rechnern machen können.

Gern hätten wir Sie erschöpfend über alle Möglichkeiten des File-Transfers zwischen der Großrechner- und der PC-Welt informiert. Da hier nicht noch eine Fortsetzungs-Story entstehen sollte – wie die über ISPF, die in dieser Ausgabe ihren Abschluß findet – möchten wir unsere Leser auf eines der nächsten Hefte vertrösten. Sie finden aber im Artikel über das Terminalprogramm GVT in der Rubrik RUM-Graphik eine Entschädigung. Dieser Artikel ist nicht nur für „Graphiker“ geschrieben, sondern für alle, die ihren PC als Terminal über eine Protokollkonverterleitung am Großrechner angeschlossen haben. GVT bietet nämlich neben den graphischen Möglichkeiten auch eine Komponente zum File-Transfer zwischen PC und CMS.

Dem programmatischen Artikel über die erste Ausbaustufe eines lokalen Rechnernetzes in der Universität ist eine Skizze beigegeben, die mit unserem Scanner S200PC erfaßt wurde, demselben, der seit dem letzten **inforum** die Redaktionsköpfe zu Papier zu bringen gestattet.



Allen Benutzern, die sich durch die Abschaffung des CMS-Kommandos QUERY NAMES vor ein paar Monaten in ihrem Kommunikationsdrang eingeschränkt fühlten, wird jetzt endlich mit QN ein Ersatz angeboten. Einzige Voraussetzung: wenn Sie bisher noch ohne NAMES-Datei auskamen, jetzt brauchen Sie sie.

Der Durchlässigkeit zwischen verschiedenen Betriebssystemen dient nicht nur der GVT-Artikel, sondern auch zwei Artikel aus dem mathematischen Bereich: Für UNIX-Benutzer steht jetzt auch eine NAG-Bibliothek in FORTRAN zur Verfügung, und Anwender des Formelmanipulationssystems REDUCE können ihre Programme auch im MVS-Batch laufen lassen. Die FORTRAN-Versionen der Präsentationsgraphik-Programme, bisher nur in PL/I „on top of GKS“ verfügbar, können jetzt auch in GKS-Anwendungen aus FORTRAN-Programmen benutzt werden.

Wem die Bindung des SCRIPT an die deutsche Sprache nicht zusagte, kann sich nun auch anderer Sprachen bedienen. Dies erlaubt der GML-Befehl *language*. Manchem mögen die vielen Amerikanismen im DV-Chinesisch schon immer aufgefallen sein. Lag's nun an den Hundstagen oder nicht, daß unsere Kollegin Annette sich einmal ihren Ärger von der Seele schreiben mußte? Wer hier noch eigene Erfahrungen mit Sprach-Veranstaltungen beisteuern möchte: Annette und die Redaktion sammeln weiter.

Als Würze in der faden DV-Suppe sei auch die Fortsetzung des von manchem vielleicht lang erwarteten **inforum**-Quiz (mit Auflösung des letzten Quizzes) im RUM-Aroma verstanden: Hätten Sie gewußt, daß die in PL/I nicht existierende Funktion FALSIFY sich mittels und ausdrücken läßt?

RUM-Aktuell

Netz-Memorandum der DFG

von
W. Held

Die Deutsche Forschungsgemeinschaft (DFG) hat in ihrem kürzlich herausgegebenen Netz-Memorandum eine Empfehlung zur Rechner-Vernetzung der Hochschulen ausgesprochen. Darin wird eine flächendeckende und leistungsfähige Netz-Infrastruktur als Grundausstattung innerhalb der Hochschulen mit Verbindungen zu anderen Hochschulen und Forschungseinrichtungen empfohlen.

Der Vorschlag sollte in allen Universitäten in den nächsten sechs Jahren realisiert sein. Jeweils etwa 10 bzw. 50 Hochschulangehörige (einschließlich Studenten) der traditionell DV-intensiven bzw. der weniger DV-intensiven Fachrichtungen sollten sich einen Netzanschluß (Schätzkosten ca. 2500 DM/Stück) teilen. Als traditionell DV-intensiv werden vor allem mathematisch-naturwissenschaftliche Fächer genannt.

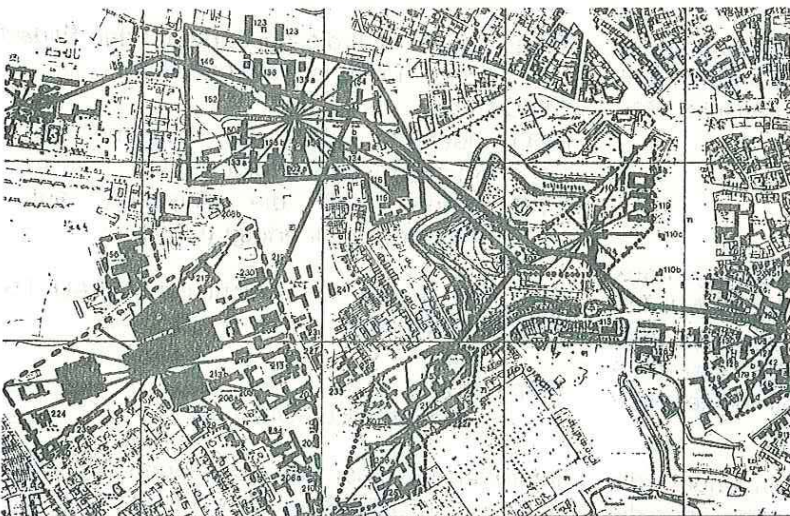
Das Rechenzentrum unserer Universität ist in diesem Bereich seit einigen Jahren besonders aktiv. Mehrere Teilnetze sind bereits im Einsatz. Wir haben daher bereits im Vorjahr zusammen mit dem Staatshochbauamt und den Dezernaten T und 4 ein Vernetzungskonzept für die gesamte Universität erarbeitet. Die Universitätsleitung hat den entsprechenden HBBG-Antrag bereits im Februar

dieses Jahres dem in Nordrhein-Westfalen zuständigen Ministerium für Wissenschaft und Forschung übergeben.

Er sieht eine schrittweise Vernetzung der Universität vor (s. Abb.). Begonnen wird im Naturwissenschaftlichen Zentrum. In den Folgejahren werden dann der Schloß-Bereich, die Altstadt sowie die übrigen eingezeichneten Gebäude erschlossen. Für die Kliniken sind die Planungen noch nicht beendet.

Die Verbindungen zur Außenwelt, die zur Zeit über EARN- (das dankenswerterweise bis Ende 1987 durch die Fa. IBM finanziert wird) bzw. Datex-P-Anschlüsse hergestellt werden, sollen im Rahmen des Deutschen Forschungsnetzes (einer Einrichtung des Bundesministeriums für Forschung und Technologie) weiter ausgebaut werden. Der DV-Referent des Landesministeriums hat zusammen mit dem Universitätsrechenzentrum ein Konzept zur Finanzierung der Leitungen in Nordrhein-Westfalen erarbeitet.

Unser Ziel ist es, jedem Nutzer eines am hochschulinternen Rechnernetz angeschlossenen Arbeitsplatzcomputers die DV-Ressourcen (Daten, Programme, Rechner und periphere Einrichtungen) innerhalb und außerhalb der eigenen Universität vom Arbeitsplatz aus zu erschließen und die Netze darüber hinaus als Kommunikationsmittel zu nutzen.



Da wir in unserer bisherigen Entwicklung im Vergleich zu den meisten anderen Hochschulen relativ weit fortgeschritten und die Planungen abgeschlossen sind, sehen wir unsere Bemühungen nun durch die Deutsche Forschungsgemeinschaft bestätigt.

QN – ein Ersatz für QUERY NAMES

von

R. Perske

Aus Datenschutzgründen mußte das CP-Kommando QUERY NAMES gesperrt werden. Als Ausgleich steht nun das Kommando QN zur Verfügung, das Ihnen anhand Ihrer NAMES-Datei die gewünschte Auskunft gibt und dabei einige komfortable Möglichkeiten bietet.

Der Aufruf

Aufgerufen wird dieses EXEC mit dem Kommando
QN (optionen

Werden dabei unzulässigerweise Operanden angegeben, so wird die zugehörige HELP-Datei auf dem Bildschirm angezeigt.

Die Ausgabe

Die Ausgabe erfolgt je nach Wahl der Option auf den Bildschirm oder in den Programmstack. Unabhängig davon kann man zwei Formate wählen: Im langen Format erfolgt die Ausgabe in der Form

```
name = userid - terminal:
      Name;Telefon
```

Im kurzen Format erscheint nur der Ausschnitt

```
userid - terminal
```

Anstelle der Terminalnummer terminal wird bei im Hintergrund arbeitenden Maschinen „disconn“, bei nicht arbeitenden Maschinen „logoff“ ausgegeben.

Die Ausgabe erfolgt sortiert: Erstes Kriterium ist, in welchem Zustand sich die Maschine des Benutzers befindet, zweites Kriterium die alphabetische Reihenfolge der Rufnamen.

Die Optionen

Es gibt drei Gruppen von Optionen, diejenigen aus den ersten beiden können mit dem DEFAULTS-Kommando voreingestellt werden. Die Optionen dürfen abgekürzt werden, dabei muß aber mindestens der groß geschriebene Teil eingegeben werden.

Die erste Gruppe beschreibt die Bedingungen, die erfüllt sein müssen, damit Informationen über die Benutzer angezeigt werden.

LOGON bzw.

NOLOGON bewirken, daß nur aktive Benutzer (wie nach LOGON) angezeigt bzw. nicht angezeigt werden. LOGON ist voreingestellt.

DISCONN bzw.

NODisconn bewirken, daß nur im Hintergrund arbeitende Benutzer (wie nach DISConn) angezeigt bzw. nicht angezeigt werden. NODISCONN ist voreingestellt.

LOGOFF bzw.

NOLOGOFF bewirken, daß nur abwesende Benutzer (wie nach LOGoff) angezeigt bzw. nicht angezeigt werden. NOLOGOFF ist voreingestellt.

Die zweite Gruppe bestimmt, woher die Informationen kommen und in welcher Form sie wohin ausgegeben werden.

FILE filename macht die Datei „filename NAMES * “ zu der nach Benutzern zu durchsuchenden NAMES-Datei.

FILE * Die NAMES-Datei „userid NAMES * “ wird nach den zu untersuchenden Benutzern durchsucht. „FILE * “ ist voreingestellt

Clear bzw.

NOClear bewirken bei der Ausgabe auf den Bildschirm, daß dieser vorher gelöscht bzw. nicht gelöscht wird. NOCLEAR ist voreingestellt

LONG bzw.

Short bewirken die Ausgabe im langen bzw. kurzen Format. Die Formate sind oben beschrieben LONG ist voreingestellt.

TYPE bewirkt, daß die Ausgabe auf den Bildschirm erfolgt. TYPE ist voreingestellt

STACK und

FIFO bewirken, daß die Ausgabe auf den Programmstack erfolgt (FIFO).

LIFO bewirkt, daß die Ausgabe auf den Programmstack erfolgt (LIFO)

Die dritte Gruppe kann nicht mit DEFAULTS voreingestellt werden:

Help und

? rufen die HELP-Datei auf.

Informationen über das Kommando QN finden Sie auch in den NEWS und als HELP-Datei, letzteres können Sie wie gewohnt mit HELP QN oder auch

mit QN ? aufrufen. Sollten bei der Benutzung dieses Kommandos Probleme auftreten, setzen Sie sich bitte mit mir in Verbindung; meine Benutzerkennung ist URZ90.

Beschränkungen auf der C-Maschine

von

B. Neukäter

In der letzten Zeit ist auch die C-Maschine in der Hauptarbeitszeit stark ausgelastet. Aus diesem Grunde sind wir im Interesse aller Benutzer gezwungen, die Rechenzeit (CPU-Zeit) und die Größe des virtuellen Hauptspeichers für jeden Benutzer ähnlich wie auf der A-Maschine zu begrenzen.

Sollten Ihre Arbeiten durch diese Beschränkungen stark behindert werden, so setzen Sie sich bitte mit mir in Verbindung (URZ20, Tel. 2689).

Das USERNAME-Kommando

von

K.-B. Mertz

Wer das CMS-Kommando „SENDJOB“ benutzt, um seine JOB-Anweisungen für MVS-Jobs erzeugen zu lassen, bekommt dort vielleicht einen Benutzernamen eingetragen, der nichts mit dem wirklichen eigenen Namen zu tun hat, sondern etwa 'QUIT' oder 'MACRO EXECUTE' lautet. Das ist in den meisten Fällen das Resultat eines falschen Tastendrucks infolge der Überraschung, die beim ersten Aufruf des SENDJOB-Kommandos durch die Frage nach dem Namen („Enter your name, please“) hervorgerufen wurde. Der zu diesem Zeitpunkt eingegebene „Benutzername“ wird gespeichert und für zukünftige CMS-Sitzungen festgehalten.

Eine Möglichkeit, diese Eintragung zu kontrollieren und zu korrigieren, wird jetzt durch das CMS-Kommando „USERNAME“ geboten. Nähere Informationen über dieses Kommando bekommen Sie durch „USERNAME ?“ oder (etwas ausführlicher) durch „HELP USERNAME“.

Kommentare oder Rückfragen richten Sie bitte an mich (URZ02, Tel. 2681).

Reduzierung der RSCS-Meldungen

von

K.-B. Mertz

Wen haben sie nicht schon gestört, die Meldungen der RSCS-Maschinen, wenn Sie Dateien an einen anderen Knoten im Netzwerk (EARN) übertragen wollen

„FILE ... ENQUEUED ON LINK DMSWWU2B“
oder übertragen haben

„SENT FILE ... ON LINK ... TO“

Das gilt wohl besonders für Jobs, die im MVS bearbeitet werden sollen, Paßwort-Änderungen, JOUTPUT- und SENDMVS-Anforderungen, bei denen man zusätzlich noch die Empfangsbestätigung des MVS/JES2 bekommt. Und beim Verschicken von NOTES oder anderen Dateien an entferntere Knoten meldet sich jede Zwischenstation, soweit es sich um einen RSCS-Knoten handelt, und unterbricht möglicherweise die XEDIT-Sitzung, in der man nur mal eben „weiterblättern“ wollte...

Ab sofort wird diese meist unerwünschte Informationsflut reduziert und nur auf Wunsch ausgelöst, weil jetzt hier wie bereits seit längerer Zeit bei den meisten anderen RSCS-Knoten im EARN/BITNET die Möglichkeit des „quiet file transfer“ installiert wurde und die Prozeduren zum Verschicken von Dateien entsprechend verändert wurden.

Bei folgenden Kommandos wird auf die RSCS-Meldungen vollständig verzichtet, weil das MVS beim Empfang der Datei eine Meldung schickt:

```
ARCHIV
JOUTPUT
JPASS (wird aufgerufen
        vom Kommando PASSWORD)
SENDMVS
```

Außerdem werden die „Acknowledgements“ für empfangene oder gelöschte Dateien (Kommandos RECEIVE und DISCARD aus dem RDRLIST-Menü oder nach PEEK) nur noch ohne RSCS-Meldungen verschickt. Schließlich erfolgt die regelmäßige (halbstündliche) Übertragung von Drucker-Dateien aus der C-Maschine, an die ja kein Drucker angeschlossen ist, zur A- oder B-Maschine ohne RSCS-Meldungen.

Bei den Kommandos SENDFILE, SENDJOB und NOTE ist ab sofort als Option „Quiet“ oder „NOQuiet“ möglich, wobei implizit „Quiet“ ange-

nommen wird, also die RSCS-Meldungen unterdrückt werden.

In Vorbereitung sind die gleichen Optionen für die Kommandos FPRINT, RPRINT und WPRINT (das ist ein XEDIT-Kommando) sowie SCRIPT. Ebenso wird noch geprüft, ob diese Optionen mit dem Kommando DEFAULTS fest eingestellt werden können.

Rückfragen und Fehlermeldungen richten Sie bitte an mich (URZ02, Tel. 2681).

Skripten

Das Benutzerhandbuch VM/CMS (Software-Information 11) ist überarbeitet worden. Diese Überarbeitung enthält die inzwischen neu eingeführten Kommandos und beschreibt die Erweiterungen bereits bestehender Befehle. Der Benutzer findet hier insbesondere auch neue Hinweise zur Datensicherung, zur Kommunikation mit fremden Maschinen usw..

Das XEDIT-Handbuch ist im März in zweiter, erweiterter Auflage erschienen.

Voraussichtlich in den Semesterferien wird das VDN-Handbuch (Software-Information 14) unter dem Titel „SQL-Handbuch“ in veränderter Form neu erscheinen.

Zusätzliche Steuerkarten bei SPSSX

von

S. Zörkendörfer

Durch Aufruf der katalogisierten Prozedur SPSSX wird im Stapelbetrieb MVS das „Statistical Package for the Social Sciences“ SPSS^X Release 2.2 aktiviert. Mit der katalogisierten Prozedur sind Steuer-Anweisungen des Betriebssystems (Job Control Language) für Regelanforderungen vorgegeben. In diesem Artikel werden vier Beispiele für Anwendungen vorgestellt, deren Betriebsmittelanforderungen darüber hinausgehen.

SPSS-Anwender unserer Installation können sich diese Beispiele mit zusätzlichen Erläuterungen durch das SPSS^X-Kommando „INFO LOKAL“ anzeigen lassen. (Die Dokumentation der besonderen Anforderungen bei der USERPROC CLUSTER erhalten Sie durch „INFO RUMCLUSTER“.)

```
//ABC01JB1 JOB (TEST,I01),DEMOSKOP,PASSWORD=.....
/*JOBPARM T=4
// EXEC RUMSERV
  ALLOCATE DAY.DEMO TRACKS(100 20) TYPE(SPSS)

//ABC01JB2 JOB (TEST,I01),DEMOSKOP,PASSWORD=.....
/*JOBPARM T=4,TAPES=1
// EXEC SPSSX
//EINGABE DD UNIT=TAPE,VOL=SER=123456,DSN=ABC01.VIELE.DATEN,DISP=OLD
//SPSSFILE DD DSN=ABC01.DAY.DEMO,DISP=(OLD,KEEP)
DATA LIST FILE=EINGABE/ TEXT 1-10 (A)
COMMENT Platz fuer Dateiaufbereitungsbefehle (Transformationen u.a.)
SAVE OUTFILE=SPSSFILE
FINISH

//ABC01JB3 JOB (TEST,I01),DEMOSKOP,PASSWORD=.....
/*JOBPARM T=4
// EXEC SPSSX
//SPSSFILE DD DSN=ABC01.DAY.DEMO,DISP=SHR
GET FILE=SPSSFILE
LIST /* oder andere SPSS-Prozedur(en)
FINISH
```

Beispiel 1.

Anwendung 1

Im ersten Job wird auf einer Magnetplatte Speicherplatz für eine Tagesdatei angefordert. Der zweite Job speichert dort ein „system file“ ab, eingelesen werden dabei „Rohdaten“ von einem Magnetband. (Beide Aufträge lassen sich natürlich auch in einem Job mit zwei Jobsteps formulieren;

evtl. sollte – zumal wenn es sich nicht um eine Tagesdatei handelt – ein weiterer RUMSERV-Jobstep folgen, der mit RELEASE den zuviel angeforderten Speicherplatz wieder freigibt.) Im dritten Job (und evtl. vielen weiteren Jobs) wird lesend auf das aufbereitete System-File zugegriffen. Die Anweisungen sind im Beispiel 1 aufgeführt.

```
//ABC02SPC JOB (TEST,I01),UMFANGREICH,PASSWORD=.....
/*JOBPARM T=4
// EXEC SPSSX
//SYSUT2 DD SPACE=(6000,200)
/* erhoehte Speicheranforderung fuer SYSUT2: 200 Bloecke zu 6000 Byte
INPUT PROGRAM
NUMERIC X1 TO X124 ZAEHLER
COMMENT : pro Fall 125 Variablen zu je 8 Byte, also 1 Kilobyte
LOOP ZAEHLER=1 TO 1000
COMMENT : also 1000 Faelle , ca. 1 Megabyte Speicherbedarf
END CASE
END LOOP
END FILE
END INPUT PROGRAM
LIST / VARIABLES=ZAEHLER/ CASES BY 100 /* ERSTE PROZEDUR
CONDESCRIPTIVE ZAEHLER /* ZWEITE PROZEDUR
FINISH
```

Beispiel 2.

```
//ABC03RGN JOB (TEST,I01),VERSCHWEND,PASSWORD=.....
/*JOBPARM T=4
// EXEC SPSSX,REGIONE=1536K
//SYSUT4 DD SPACE=(6000,100)
INPUT PROGRAM
NUMERIC X Y X3 TO X180
LOOP X=1 TO 10
COMPUTE Y=X*X
END CASE
END LOOP
END FILE
END INPUT PROGRAM
REGRESSION / MISSING=PAIRWISE / VARIABLES=X TO X180/
DEPENDENT=X / METHOD=ENTER Y
COMMENT grosser Hauptspeicherbedarf: 180*180 Korrelationsmatrix
ferner Zwischenspeicherung auf Plattendatei SYSUT4
FINISH
```

Beispiel 3.

Anwendung 2

In dieser Anwendung ist für die SYSUT2-Datei mehr Speicherbedarf erforderlich als von der

katalogisierten Prozedur vorgegeben. Auf SYSUT2 werden die Variablenausprägungen abgespeichert – hier wird es also des öfteren zu Kapazitätsproblemen kommen. Es wird empfohlen, gezielt die

SPACE-Anforderung für SYSUT2 heraufzusetzen – eine Abschätzung des benötigten Speicherplatzes sollte vorausgegangen sein. (Eine Benutzung des CYL-Parameters ist i.d.R. ungeeignet, weil damit der Speicherbedarf für viele Dateien heraufgesetzt

wird – der für die temporären virtuellen Dateien auf UNIT=WORK insgesamt verfügbare Speicherplatz ist jedoch auf 10MB beschränkt.) Sie finden die zugehörigen Anweisungen im Beispiel 2.

```
//ABC04MCH JOB (TEST,I01),PAPIERFRASS,PASSWORD=(<PASSWORD>)
/*JOBPARM T=4,L=10
// EXEC SPSSX,EPRINT=A,EHOLD=YES
/** Zur Demonstration der "JOUTPUT"-Anweisung des RUM im CMS
/**      jstatus      mch
/**      joutput get   mch (from 1 for 111) ; rl ; peek
/**      joutput get   mch (last 111      ) ; rl ; peek
/**      joutput print mch      |      joutput purge mch
/** Weiter Informationen unter CMS durch "HELP JOUTPUT"
COMMENT es wird eine umfangreiche Ausgabe erwartet;
        bitte nicht Andrucken, falls etwas schiefgeht!

:
:
FINISH
```

Beispiel 4.

Anwendung 3

Bei nur wenigen Prozeduren wird es zu erhöhter Anforderung an Hauptspeicherbedarf kommen, wie in Beispiel 3 angegeben.

Anwendung 4

Einige Hinweise zum CMS: In der Regel wird ein SPSS^X-Auftrag unter dem Dialog-Betriebssystem

CMS erstellt und an das Stapel-Betriebssystem MVS übergeben („sendjob“). Die Ausgabe wird dann auf den „Reader“ der VM-Maschine geschickt, kann dort kontrolliert und von dort zum Drucker geleitet werden. Umfangreiche Druckausgabe sollte jedoch nicht vom MVS zum CMS geleitet werden, deshalb weise ich auf das Kommando JOUTPUT hin (vgl. Beispiel 4).

Personalia

Als studentische Mitarbeiter wurde zum 1. 4. 87 Herr Myong Joong Kim eingestellt.

Ausgeschieden sind die folgenden studentischen Mitarbeiter: Herr Martin Brox zum 30. 4. 87 und Herr Dieter Stüken zum 31. 5. 87.

Mit Wirkung vom 1. 6. 87 ruht das Arbeitsverhältnis von Herrn Peter Bigdon für die Dauer seiner 15-monatigen Zivildienstzeit. Herr Bigdon hat sich um die Textverarbeitung bei der Herausgabe des Inforum Verdienste erworben, wofür ihm die Redaktion ihren Dank ausspricht.

Unserem Redaktionsmitglied Walter Bosse wünschen wir für seine baldige Genesung alles Gute.

REDUCE im MVS verfügbar

von

B. Süsselbeck

Ab sofort ist das Formelmanipulationssystem REDUCE auch im MVS verfügbar. Der Aufruf erfolgt durch die katalogisierte Prozedur REDUCE:

```
// EXEC REDUCE
```

Speicherplatzanforderungen werden über den Parameter REGIONE übergeben. Wenn Sie nähere Auskünfte brauchen, wenden Sie sich bitte an mich (URZ22, Tel. 2686).

NAG-Bibliothek im UTS

von

B. Süselbeck

Im UTS steht jetzt die NAG-Bibliothek in doppelter Genauigkeit zur Verfügung. Ist „beisp.f“ eine Datei, die ein FORTRAN-Programm enthält, das Routinen aus der NAG-Bibliothek aufruft, so wird durch das Kommando

```
f77 beisp.f -L/rumsys/lib -lnag
```

ein Lademodul erzeugt, der mit dem Kommando

```
a.out
```

zur Ausführung gebracht werden kann.

Hinweise:

- Die Linkage-Editor-Optionen -L und -I müssen mit ihren Argumenten beim Aufruf von f77 hinter dem Namen der Datei, die das FORTRAN-Programm enthält, angegeben werden.
- Die Argumente der Optionen müssen ohne Leerzeichen direkt an die Buchstaben L bzw. I angehängt werden.
- Die „-L“-Option muß vor der „-I“-Option angegeben werden.

Bei Nachfragen wenden Sie sich bitte an mich (URZ22, Tel. 2686).

Ein neuer GML-Befehl – :language

von

R. Perske

Von verschiedenen Benutzern des Textformatierungssystems SCRIPT/VS wurde zu Recht angemerkt, daß es sehr schwierig sei, Texte in anderen Sprachen als Deutsch abzufassen, da vom System eingefügte Phrasen wie „Abbildung“, „Tabelle“, „Inhaltsverzeichnis“ oder „siehe auf Seite“ immer der deutschen Sprache entnommen werden. Zwar gibt es die Möglichkeit, die jeweiligen Variablen umzudefinieren, doch ist diese Methode sehr mühsam und verlangt genauere Kenntnisse des Systems. Diese Arbeit übernimmt jetzt der neue GML-Befehl :language, der in diesem Artikel beschrieben werden soll.

Aufruf

Der Befehl wird in der Form

```
:language attribut.
```

aufgerufen. Für attribut ist eines der in der unten stehenden Tabelle angegebenen Schlüsselwörter einzusetzen. Wird ein ungültiges Attribut angegeben, so erfolgt eine Fehlermeldung der Schwere /E/. SCRIPT setzt in diesem Falle die Verarbeitung nur dann fort, wenn die Option Continue gesetzt ist.

Der Aufruf des Befehls darf an beliebiger Stelle des Textes erfolgen, es ist also durchaus möglich, mitten im Text die Sprache zu wechseln. Implizit wird im Script-Vorspann der Aufruf

```
:language deutsch.
```

durchgeführt.

Es empfiehlt sich, bei einem nicht deutschsprachigen Text den Befehl :language zu Beginn des Textes zu verwenden, z.B. unmittelbar nach :gdoc.

Funktion

Der Befehl :language schaltet die gesamte laufende Verarbeitung auf die gewünschte Sprache um, das betrifft:

- Rechtschreibprüfung (soweit beim Aufruf des SCRIPT-Kommandos angegeben)
- Silbentrennung
- Eingefügte Phrasen (Beispiele s.o.)
- Form von Datum und Zeitangabe (Variablen &date. und &time.)
- Form der Anführungszeichen (bei Benutzung der Befehle :q. und :eq.)

Sprachen

Welche Sprachen zur Verfügung stehen und welche Schlüsselwörter diese Sprachen repräsentieren, entnehmen Sie bitte der unten abgedruckten Tabelle. Voreingestellt ist die deutsche Sprache, ausgewählt werden können auch noch die englischen Sprachvarianten. Die anderen aufgeführten Sprachen sind bereits vorgesehen, konnten aber aufgrund der mangelnden Sprachkenntnisse des Autors noch nicht fertiggestellt werden. Wer eine

andere dieser Sprachen benötigt, sollte sich an Herrn Neukäter (URZ20) oder an mich (URZ90) wenden.

Attribute zu :language

Folgende Tabelle enthält die Attribute, mit der die jeweilige Sprache eingestellt werden kann.

Sprache	Attribute
Deutsch	germ german deutsch
Englisch (USA)	eam englisham englishus
Englisch (GB)	euk englishuk english
Englisch (CAN)	ecan englishcan
Französisch (F)	fnat french
Französisch (CAN)	fcen frenchcan
Niederländisch	dtch dutch
Italienisch	ital italian
Spanisch	span spanish

RUM-Lehre

Lehrveranstaltungen im 2. Halbjahr 1987

In vielen Bereichen ist die Datenverarbeitung zu einem unerläßlichen Hilfsmittel für wissenschaftliches und praktisches Arbeiten geworden. Daher wird den Studenten – nicht nur aus der Mathematik und den experimentell oder empirisch ausgerichteten Disziplinen – dringend empfohlen, sich bereits in den ersten Studiensemestern Grundkenntnisse in der EDV, insbesondere in der Programmierung, anzueignen.

Zu den Ferienkursen

Im Anschluß an die Vorlesungszeit im SS 1987 sowie vor Beginn des WS 1987/88 werden vom Universitätsrechenzentrum einige ganztägige Intensivkurse durchgeführt, in denen Stoffvermittlung und Übungen integriert sind. Diese Veranstaltungen sollen durch entsprechende Betreuung der Teilnehmer eigene Programmierübungen fördern. Das bedingt eine Begrenzung der Teilnehmerzahl der einzelnen Veranstaltungen. Interessenten werden deshalb gebeten, sich möglichst bald, spätestens jedoch eine Woche vor Beginn der entsprechenden Veranstaltung, im Dispatch des Rechenzentrums (Raum 02) in die Anmelde-listen einzutragen, und sollten unbedingt zu dem angekündigten Beginn anwesend sein.

Die Teilnehmer dieser Kurse werden gebeten, diese im WS 1987/88 zu belegen. Dies ist neben der bestandenen Abschlußklausur Voraussetzung für die Aushändigung eines Scheines über die erfolgreiche Teilnahme.

Zu den Semesterkursen

Eine Anmeldung ist nur für diejenigen Lehrveranstaltungen erforderlich, die nachfolgend besonders gekennzeichnet sind.

1. Veranstaltungen in den Semesterferien (Juli – Oktober 1987)

320016	Programmieren in FORTRAN vom 20.07. bis 31.07.1987 ganztägig Hörsaal: M4, Beginn: 20.07.1987, 9 Uhr	Reichel, K.
320020	Programmieren in FORTRAN vom 24.08. bis 04.09.1987 ganztägig Hörsaal: M3, Beginn: 24.08.1987, 9 Uhr	Stöckelmann, D.
320035	Programmieren in Pascal vom 31.08. bis 11.09.1987 ganztägig Hörsaal: M5, Beginn: 31.08.1987, 9 Uhr	Achilles, A.
320040	Programmieren in Pascal vom 28.09. bis 09.10.1987 ganztägig Hörsaal: M3, Beginn: 28.09.1987, 9 Uhr	Steinhausen, D.
320054	Programmieren in PL/I vom 07.09. bis 25.09.1987 ganztägig Hörsaal: M4, Beginn: 07.09.1987, 9 Uhr	Mersch, R.
320069	Programmieren in C vom 10.08. bis 21.08.1987 ganztägig Hörsaal: M4, Beginn: 10.08.1987, 9 Uhr	Kämmerer, M.

- 320073 Statistische Datenanalyse mit dem Programmsystem SPSS^X Nienhaus, R.
vom 28.09. bis 09.10.1987 ganztägig
Hörsaal: M4, Beginn: 28.09.1987, 9 Uhr

2. Einführende Veranstaltungen im Wintersemester 1987/88

- 320088 Programmieren in FORTRAN Mertz, K.-B.
Fr 13 – 15
Hörsaal: M4, Beginn: 16.10.1987
- 320092 Programmieren in FORTRAN Schalthöfer, H.
Mi 13 – 15
Hörsaal: M4, Beginn: 14.10.1987
- 320107 Programmieren in Pascal Süselbeck, B.
Di 13 – 15
Hörsaal: M4, Beginn: 20.10.1987
- 320111 Programmieren in Modula-2 Pudlatz, H.
Mi 15 – 17
Hörsaal: M4, Beginn: 21.10.1987
- 320126 Programmieren in PL/I Ost, St.
Mo 15 – 17
Hörsaal: M5, Beginn: 19.10.1987
- 320130 *) Textverarbeitung auf Mikrorechnern Kamp, H.
Do 13 – 15
Hörsaal: M5, Beginn: 22.10.1987
- 320145 Computerunterstütztes Publizieren mit L^AT_EX Kaspar, W.
Mi 9 – 11
Hörsaal: M4, Beginn: 28.10.1987
- 320150 Das Statistical Analysis System SAS Steinhausen, D./
Mo 15 – 17 Zörkendörfer, S.
Hörsaal: M4, Beginn: 19.10.1987
- (121676) *) English for Computer Science Gress, E.
Mo 13 – 15
Hörsaal: M6

*) Wegen der Begrenzung der Teilnehmerzahl ist für diese Lehrveranstaltung eine frühzeitige Anmeldung im Dispatch des Rechenzentrums erforderlich.

3. Weiterführende Veranstaltungen im Wintersemester 1987/88

- 320164 Datenstrukturen und Programmierverfahren in Pascal Bosse, W.
Di 16 – 18
Hörsaal: M2, Beginn: 20.10.1987
- 320179 Programmiertechniken Neukäter, B.
Mi 17 – 19
Hörsaal: M4, Beginn: 14.10.1987

320183	Einführung in die Programmiersprache C Do 15 – 17 Hörsaal: M5, Beginn: 22.10.1987	Lange, W.
320198	Graphische Datenverarbeitung Do 15 – 17 Hörsaal: M4, Beginn: 22.10.1987	Sturm, E.
320202	Mikrorechner Mo 13 – 15 Hörsaal: M4, Beginn: 19.10.1987	Kisker, H.-W.
320217	Unix Mi 11 – 13 Hörsaal: M4, Beginn: 28.10.1987	Stockhofe, W.
320221	Rechnernetze Do 13 – 15 Hörsaal: M4, Beginn: 29.10.1987	Richter, G./ Schachtner, G.
320236	Kolloquium über Themen der Informatik Fr 15 – 17 Hörsaal: M4	Held, W./ die wiss. Mitarbeiter des Rechenzentrums
320240	Anleitung zum Einsatz der EDV bei wissenschaftlichen Arbeiten	die wiss. Mitarbeiter des Rechenzentrums

Kommentare zu den Lehrveranstaltungen

Programmieren in FORTRAN

(320016, 320020, 320088, 320092)

FORTRAN ist eine Programmiersprache, die vorwiegend für die Formulierung von Problemlösungen aus dem Bereich der Naturwissenschaften (Numerik, Statistik) geeignet ist. Leichte Erlernbarkeit und Anwendbarkeit auf Computern fast aller Hersteller haben dieser Programmiersprache eine weite Verbreitung gesichert.

Literatur:

VS-FORTRAN Release 4.0 (erhältlich im Rechenzentrum)

Lamprecht: Einführung in die Programmiersprache FORTRAN 77, Vieweg

Wehnes: FORTRAN 77, Hanser

Kießling/Lowes: Programmierung mit FORTRAN 77, Teubner Studienskripten

Brauer: Programmieren in FORTRAN 77, Hüthig

Gibson/Young: Introduction to Programming Using FORTRAN 77, Prentice Hall

Programmieren in Pascal

(320035, 320040, 320107)

Pascal ist eine moderne Programmiersprache und aufgrund ihres didaktischen Konzepts für Programmieranfänger besonders zu empfehlen. Andererseits ist Pascal durch die Vielzahl von Datentypen und Strukturierungsmöglichkeiten für vielfältige Anwendungen numerischer wie nichtnumerischer Art gleichermaßen geeignet. Die Sprache ist auch dadurch interessant geworden, daß sie auf Mikrorechnern einsetzbar ist.

Literatur:

Marty: Methodik der Programmierung in Pascal, Hanser

Jensen/Wirth: Pascal, User Manual and Report, Springer

Ottmann/Widmeyer: Programmieren mit Pascal, Teubner

Wilson/Addyman: Pascal, Leichtverständliche Einführung, Hanser

Datenstrukturen und Programmierverfahren in Pascal (320164)

In dieser weiterführenden Lehrveranstaltung werden insbesondere dynamische Datenstrukturen (Stack, Liste, Baum u.a.) sowie Fragen der Programmiermethodik anhand praktischer Beispiele behandelt. Schwerpunkte bilden dabei effiziente Sortierverfahren und Aufgaben der Listenverarbeitung. Im Hinblick auf die Erstellung leistungsfähiger und übersichtlicher Programme sollen bereits vorhandene grundlegende Programmierkenntnisse in Pascal vertieft werden.

Literatur:

Wirth: Algorithmen und Datenstrukturen, Teubner
Schneider/Bruell: Advanced Programming and Problem Solving with Pascal, John Wiley & Sons

Programmieren in PL/I (320054,320126)

PL/I ist die universelle Programmiersprache für IBM-Rechner. Ihre Anwendungsmöglichkeiten reichen von der Textverarbeitung über die Programmierung numerischer Probleme bis zur Bearbeitung großer Datenmengen mit Hilfe vielseitiger Ein/Ausgabebefehle. Dynamische Speicherplatzverwaltung und selbst Multitasking, also der gleichzeitige Ablauf mehrerer Unterprogramme, sind in PL/I möglich.

Im Grundkurs können selbstverständlich nicht all diese Sprachelemente vorgestellt werden. Im Vordergrund steht die Vermittlung moderner Programmiermethoden anhand leichtverständlicher Beispiele, deshalb vorwiegend aus dem nichtnumerischen Bereich. Hierzu stellt PL/I alle Werkzeuge strukturierter Programmierung zur Verfügung.

Literatur:

Conway/Gries: An Introduction to Programming, Winthrop
Conway/Gries: Der neue Programmierstil, gezeigt an PL/I, Hanser
Mell/Preuß/Sandner: Einführung in die Programmiersprache PL/I, BI
Kamp/Pudlatz: Einführung in die Programmiersprache PL/I, Vieweg

Programmieren in C (320069)

C ist eine Programmiersprache, deren Einsatzmöglichkeiten einerseits durch Assembler-ähnliche Sprachelemente und andererseits durch Elemente moderner blockstrukturierter Sprachen sehr vielseitig sind. Durch einen hohen Grad an Portabilität ist C eine Sprache mit stark zunehmender Verbreitung; Compiler-Implementierungen stehen praktisch für alle Betriebssysteme und Rechnertypen zur Verfügung.

Die Lehrveranstaltung ist vorrangig für Hörer mit Programmiererfahrung in anderen Sprachen bestimmt.

Literatur:

Harbison/Steele: C Reference Manual, Prentice Hall
Kernighan/Ritchie: Programmieren in C, Hanser

Programmieren in Modula-2 (320111)

Modula-2 ist eine von N. Wirth um 1980 vorgestellte Weiterentwicklung der Sprache Pascal. Durch das „Modul“-Konzept, das die Trennung in einen von außen sichtbaren Definitionsteil und einen von außen unsichtbaren Implementationsteil erlaubt, wird die Programmierung komplexer Systeme im Team unterstützt. Multiprogramming ist möglich („Prozeß“), und Programmkonstrukte auf niedriger Ebene ermöglichen die Umgehung des von Pascal her bekannten „strong typing“. Als weiterer Datentyp ist der Procedure-Typ vorhanden, der die Zuweisung von Prozeduren an Variable und damit die Übergabe von Prozeduren als Parameter erlaubt.

Modula-2 steht sowohl in einer Großrechner-Version als auch auf PCs zur Verfügung.

Vorkenntnisse einer höheren Programmiersprache werden nicht vorausgesetzt.

Literatur:

Wirth: Programming in Modula-2, Springer
Dal Cin/Lutz/Risse: Programmierung in Modula-2, Teubner
Pomberger: Softwaretechnik und Modula-2, Hanser

Programmiertechniken (320179)

Um die EDV wirksam einsetzen zu können, reichen Grundkenntnisse in einer Programmiersprache nicht aus. Auch die Beherrschung einer Programmiersprache ist noch keine Garantie, für jedes Problem der Datenverarbeitung die angemessene Lösung zu finden. Erforderlich ist ein Überblick über den Stand der Programmiertechnik und ein Einblick in die Methodik des Programmierens in verschiedenen Programmierumgebungen. In dieser Veranstaltung sollen Programmiertechniken vorgestellt werden. Zur Illustration dieser Techniken werden Beispiele in Programmiersprachen wie FORTRAN, Pascal, PL/I, Modula-2, C vorgeführt. Darüber hinaus wird anhand von APL und REXX gezeigt, wie man Anwendersysteme im Dialog rasch und präzise aus Programmbausteinen entwickeln kann und welche Programmierumgebungen zur schnellen Fehlererkennung besonders geeignet sind.

Statistische Datenanalyse mit dem Programmsystem SPSS^X (320073)

Das statistische Programmsystem SPSS (Statistical Package for the Social Sciences) wird in der am Rechenzentrum implementierten Version SPSS^X vorgestellt. Mit diesem System stehen bequem aufzurufende Programme zu den gebräuchlichen univariaten und multivariaten statistischen Verfahren sowie zur Datenmanipulation zur Verfügung. SPSS wird z.B. zur statistischen Auswertung von Fragebögen eingesetzt.

In dieser Veranstaltung wird das programmtechnische Rüstzeug zur Durchführung derartiger Auswertungen vermittelt. Solide Grundkenntnisse bezüglich der anzusprechenden statistischen Verfahren sowie Kenntnisse der Anwendungsmöglichkeiten dieser Verfahren im jeweiligen Fachgebiet sind erwünscht und bei den praktischen Übungen von großem Nutzen.

Literatur:

Steinhausen/Zörkendörfer: Statistische Datenanalyse mit dem Programmsystem SPSS^X (erhältlich im Rechenzentrum als *Software-Information 13*)

Das Statistical Analysis System SAS (320150)

Vorgestellt wird die Großrechner-Dialogversion SAS 5.16 mit SAS/FSP und SAS/GRAPH als ein mächtiges Anwendungssystem zur Datenanalyse, Dateienverwaltung und Ergebnisrepräsentation. Die Teilnehmer werden Programme in der Programmiersprache SAS erstellen, es werden vielfältige Möglichkeiten der SAS-Programmierung aufgezeigt, zum interaktiven Arbeiten und zur Anbindung an den Rechner und das zugrunde liegende Betriebssystem wird das SAS-System (mit DMS) benutzt. Ziel der Veranstaltung ist es, nach Erlernen der SAS-Programmiertechniken Lösungen für reiche Anwendungsprobleme mit sehr geringem Programmieraufwand zu erstellen.

Die Veranstaltung wendet sich an Hörer aller Fachrichtungen; neben der eigentlichen Vorlesungszeit ist Zeitaufwand für eigene praktische Übungen am Terminal einzuplanen.

Literatur:

SAS User's Guides (Basics / Statistics / FSP / Graph)

Textverarbeitung auf Mikrorechnern (320130)

Im ersten, theoretischen Teil der Veranstaltung sollen Konzeption und Realisierungsmöglichkeiten textverarbeitender Systeme besprochen werden. Im zweiten, mehr praxisorientierten Teil, soll den Teilnehmern die Gelegenheit geboten werden, ausgewählte Systeme zu testen und miteinander zu vergleichen. Die Tests sollen auf Mikrorechnern des Universitätsrechenzentrums durchgeführt werden.

Computerunterstütztes Publizieren mit LATEX (320145)

TEX (sprich: ‚Tech‘) ist ein *Satzsystem*, das innerhalb der letzten 10 Jahre von Donald E. Knuth an der Stanford University entwickelt wurde.

LATEX ist eine Erweiterung von TEX, die das Arbeiten mit TEX wesentlich erleichtert. Dem Autor werden z.B. fertige Layouts für Bücher, Reports und Artikel zur Verfügung gestellt, die er selbst in gewissen Grenzen seinen eigenen Vorstellungen leicht anpassen kann.

L^AT_EX steht sowohl auf IBM-kompatiblen PCs als auch auf unserem Großrechner zur Verfügung. Der Wechsel von einem Rechner zum anderen ist jederzeit möglich.

In dieser Veranstaltung werden die Einsatzmöglichkeiten von L^AT_EX im Publikationsprozeß vorgestellt. Es wird gezeigt, wie Texte mit Editoren wie KEDIT unter DOS und XEDIT unter CMS für L^AT_EX erfaßt, mit T_EX formatiert, zur Kontrolle am Bildschirm angezeigt und auf unterschiedlichen Druckern ausgegeben werden können.

Außerdem werden Hinweise zur Änderung der vorgegebenen Layouts und für den Übergang von anderen Textverarbeitungsprogrammen, wie z.B. DCF, gegeben.

Die Hörer sollten Grundkenntnisse im Umgang mit PCs oder dem Großrechner besitzen.

Literatur:

Lamport, L., L^AT_EX – A Document Preparation System, Reading, Mass.: Addison-Wesley (1986).

Schwarz, N., L^AT_EX Einführung in T_EX, Bonn: Addison-Wesley (1987).

English for Computer Science (121676)

Grundkenntnisse der Datenverarbeitung sollten vorhanden sein. Wir wollen uns in diesem Semester mit aktuellen Themen aus der Datenverarbeitung beschäftigen. Dabei geht es nicht darum, sich neue Fachkenntnisse anzueignen, sondern sich im Englischen zu üben. Es können Themen- bzw. Textvorschläge eingebracht werden.

UNIX (320217)

UNIX ist ein weitverbreitetes Mehrbenutzerbetriebssystem. Es ist auf Rechnern verschiedener Hersteller und unterschiedlicher Leistungsklassen ablauffähig. Damit steht dem UNIX-Anwender vom Mikrorechner bis zum Großrechner die gleiche leistungsfähige und komfortable Programmier- und Arbeitsumgebung zur Verfügung. Hardware-Unterschiede der einzelnen Maschinen werden weitgehend verdeckt.

RUM-Lehre

Literatur:

Kaare: The UNIX Operating System, Wiley

Mikrorechner (320202)

Die Vorlesung führt in die Grundlagen und die Nutzung von Mikrorechnern ein. Insbesondere werden folgende Themen behandelt:

- Grundlegender Aufbau von Mikrorechnern
- Betriebssystem MS-DOS
- Vorstellung einiger Anwendungsprogramme
- Hardware-Erweiterungen
- Systemprogrammierung von Mikrorechnern

Literatur:

Norton: MS-DOS und PC-DOS, Hanser

Norton: Die verborgenen Möglichkeiten des IBM-PC, Hanser

Rechnernetze (320221)

Die Entwicklung der Mikrocomputer zu leistungsfähigen autonomen Arbeitsplatzrechnern, der verstärkte Einsatz zahlreicher Mikrocomputer innerhalb einer Organisation und die damit einhergehende Dezentralisierung von Rechenleistung erfordert effiziente Möglichkeiten des Informationsaustausches zwischen Rechnern auch unterschiedlicher Typen und Hersteller und der Informationssammlung auf zentralen EDV-Anlagen. Dies hat die Entwicklung und Verbreitung lokaler Rechnernetze sowie deren Anbindung an bestehende, große Entfernungen überbrückende, globale Rechnernetze intensiv vorangetrieben. In der Veranstaltung soll auf logische und technische Grundlagen sowie theoretische Konzeptionen vor allem lokaler Rechnernetze eingegangen werden. Der Praxisbezug soll durch Anwendungsbeispiele aus dem Bereich der Universität hergestellt werden. Z.B. werden der Betrieb der Ausbildungsrechnernetze („CIP-Pools“), die Ansätze für ein hochschulweites Datenverbundsystem und der Zugang zu Datenbanken über öffentliche Netze behandelt werden.

Literatur:

Meißner: Arbeitsplatzrechner im Verbund, Hanser
Schnupp: Rechnernetze – Entwurf und Realisierung, de Gruyter
Hopper, Temple, Williamson: Local Area Network Design, Addison-Wesley

**Graphische Datenverarbeitung
(320198)**

Mit dem Graphischen Kern-System (ISO- und DIN-Norm GKS) ist die Schnittstelle zwischen dem Anwendungsprogramm und der systemnahen graphischen Software definiert. Ob Bilder für Plotter oder Bildschirm zu erzeugen sind, schwarz-weiß oder farbig, als Vektor- oder Raster-Graphik, im Job oder interaktiv: immer werden vom Benutzer dieselben Unterprogramme aufgerufen. In dieser Lehrveranstaltung wird sowohl auf die GKS-„Philo-

sophie“ eingegangen, als auch an konkreten Beispielen demonstriert, wie am hiesigen Rechenzentrum GKS-Programme zu schreiben sind. Zur Verfügung steht RUM/GKS, die GKS-Version des Rechenzentrums, für die Programmiersprachen PL/I und FORTRAN.

Literatur:

Bechlars, Buhtz: GKS in der Praxis, Springer
Enderle, Kansy, Pfaff: Computer Graphics Programming, Springer

**Kolloquium über Themen der Informatik
(320236)**

Im Rahmen des Kolloquiums werden Vorträge über neuere Entwicklungen der Informatik gehalten. Vortragstermine werden durch Aushang im Universitätsrechenzentrum bekanntgegeben.

RUM-Graphik

Präsentationsgraphik in FORTRAN

von

H. Pudlatz

Getreu der Devise, GKS-Entwicklungen an unserem Rechenzentrum zunächst in PL/I durchzuführen und später in FORTRAN umzusetzen, sind jetzt auch zwei Programme aus dem Bereich der Präsentationsgraphik für FORTRAN-Benutzer verfügbar gemacht worden. Es handelt sich um `DEFINE_BARCHART`, `GENERATE_BARCHART`, `DEFINE_PIECHART`, `GENERATE_PIECHART` zur Definition und Erzeugung von Balken- bzw. Tortendiagrammen (vgl. *inforum* Nr.1/1985: „Balken- und Tortendiagramme für GKS-Benutzer“).

Wie in der PL/I-Version kommt hier dem vorherigen Aufruf der Routinen zur Festlegung des Fensters im Weltkoordinatensystem („Window“) und dem des Darstellungsbereichs („Viewport“) eine wichtige Rolle zu, da ein Diagramm stets in den gewählten Viewport eingepaßt wird. Falls (eigene) Beschriftungen an einem Diagramm angebracht werden sollen, ist zum einen die Lage des Viewports in der Zeichenfläche geeignet festzulegen und zum anderen durch den Aufruf `CALL GSCLIP(0)` das Abschneiden (Clipping) von Teilen der Zeichnung am Viewport zu verhindern. Bei komplexeren Graphiken ist es empfehlenswert, den `COMMON`-Bereich `GKSCOM` zu vergrößern.

Die beiden Darstellungsarten sollen nun nacheinander erläutert werden, zunächst zum Balkendiagramm:

Ein einfaches Balkendiagramm kann man durch den Aufruf

```
CALL GENBAR (LAGE, XKOORD, YKOORD, N)
```

erhalten, wobei für die Parameter folgendes gilt:

LAGE (CHARACTER*(*))

'HORIZONTAL' = horizontale Balken,

'VERTICAL' = vertikale Balken (Normalfall)

XKOORD (REAL) X-Koordinaten der Balken

YKOORD (REAL) Y-Koordinaten der Balken

N (INTEGER) Anzahl der Elemente der vorstehenden Vektoren ($2 \leq N \leq 30$)

Der Parameter **LAGE** kann bis auf den Anfangsbuchstaben abgekürzt werden. Je nachdem, ob **LAGE** = 'H' oder 'V' ist, werden die X- bzw. Y-Koordinaten mittels der Balkenhöhen dargestellt, wobei das jeweils andere Feld die Lage der Balken bestimmt. Dieses Feld der „Basiskoordinaten“, das die Balkenmitten auf einer gedachten Basislinie festlegt, muß dann jeweils aufsteigend sortiert sein, da durch die Differenz aufeinanderfolgender Werte dieses Feldes ein positiver Abstand der Balkenmitten gegeben sein muß. In der Regel wird man für die Basiskoordinaten einen Abschnitt der Folge 1, 2, 3, ... wählen.

Will man zu einer Basiskoordinate jeweils mehrere Balken zeichnen, so ist eines der Felder **XKOORD** bzw. **YKOORD** als NxM-Matrix und das jeweils andere als N-Vektor der Basiskoordinaten zu übergeben. Hierbei werden zu jeder der N Basiskoordinaten M Werte als hintereinanderliegende Balken dargestellt. Die Subroutine ist dann mit einem zusätzlichen Parameter aufzurufen:

```
CALL GENBAR (LAGE, XKOORD, YKOORD,
              N, M)
```

Für diesen gilt:

M (INTEGER) Spaltenzahl der Wertematrix ($2 \leq M \leq 10$)

Ist **LAGE** = 'V', so muß **XKOORD** der Vektor der Basiskoordinaten und **YKOORD** die Wertematrix sein, anderenfalls ist die Bedeutung der beiden Felder zu vertauschen. Bei der Darstellung der Balken hintereinander werden die implizit bei `FILL_AREA` verwendeten Schraffurtypen 1 bis 10 zur Unterscheidung der Balken benutzt. Will man hier eigene Setzungen vornehmen, so kann man sich (analog zu `DEFINE_BARCHART`) der folgenden Subroutine bedienen:

```
CALL DEFBAR (BREIT, VERSTZ, FARBE,
              SCHRAF, N)
```

Die Breite der Balken ist implizit die kleinste Differenz der Basiskoordinaten, sie kann hier durch die Angabe von **BREIT** modifiziert werden, muß aber kleiner als der genannte Wert bleiben, um Überschneidungen zu vermeiden.

Ebenso ist es manchmal zweckmäßig, bei der Darstellung mehrerer Variablen mit GENBAR die Balken gegeneinander zu versetzen, um hintereinanderliegende Balken besser voneinander unterscheiden zu können.

Die Vektoren FARBE und SCHRAF, deren gemeinsame Dimension mit N übergeben wird, entsprechen den von FILL_AREA her bekannten Angaben FILL_AREA_COLOUR_INDEX und FILL_AREA_STYLE_INDEX zur Schraffur der Balken.

Im einzelnen bedeuten die Parameter:

BREIT (REAL) Breite der Balken

VERSTZ (REAL) Versatz der Balken bei mehrdimensionaler Darstellung (negative Werte ergeben einen Versatz nach links, positive einen Versatz nach rechts)

FARBE (INTEGER) Farbindexvektor (vgl. Setzungen bei GSFACI)

SCHRAF (INTEGER) Schraffurvektor (vgl. Setzungen bei GSFASI)

N (INTEGER) Dimension der Farb- und Schraffurvektoren

Das unten angeführte Beispiel 1 soll diesen Fall bei der Darstellung einer Matrix erläutern.

```

PROGRAM BARS
*----- Balkendiagramme ("Ergebnisse einer fiktiven Gemeinderatswahl")
CHARACTER*7 TEXT (5) /'CDU','SPD','FDP','Grüne','sonst'/
REAL WIDTH /0.6/, OFFSET /0.2/, HEIGHT /0.2/,
1 WINDOW (4) / 0., 10., 0., 50./,
2 VIEWPT (4) / 4E-2, 24E-2, 4E-2, 24E-2/,
3 X (5) / 1, 2, 3, 4, 5/,
4 PROZNT(5,2) / 42.3, 46.3, 5.2, 4.5, 1.7,
5 44.7, 41.1, 4.8, 2.8, 0.5/
INTEGER SCHRAF (5) / 3033, 4041, 3062, 3014, 3048/,
1 FARBE (5) / 1, 2, 3, 1, 2/
COMMON /GKSCOM/ GKSCOM (6000)
CALL GSPLT (6000)
CALL GSTXFP (4,2)
CALL GSCHH (HEIGHT)
CALL GSWN (1, WINDOW(1), WINDOW(2), WINDOW(3), WINDOW(4))
CALL GSPV (1, VIEWPT(1), VIEWPT(2), VIEWPT(3), VIEWPT(4))
XFAKT = (VIEWPT(2) - VIEWPT(1)) / (WINDOW(2) - WINDOW(1))
YFAKT = (VIEWPT(4) - VIEWPT(3)) / (WINDOW(4) - WINDOW(3))
CALL DEFBAR (WIDTH, OFFSET, FARBE, SCHRAF, 5)
CALL GENBAR ('V', X, PROZNT, 5, 2)
CALL DEFAXI (0.,5., 4,4,0,0, 1E-3,2E-3,2E-3,2E-3,2E-3,2E-3)
CALL GENAXI ('Y1', 'Prozente')
CALL DEFAXI (1.,1., 0,1,0,0, 1E-3,2E-3,2E-3,2E-3,2E-3,2E-3)
CALL GENAXI ('X1', 'Parteien')
CALL GSCLIP (0)
CALL GSTXFP (4,2)
CALL GSTXAL (1,3)
CALL GSCHH (YFAKT)
CALL GSCHUP (-1.,0.)
DO 10 I = 1, 5
CALL GSTXCI (FARBE(I))
10 CALL GTX (VIEWPT(1)+X(I)*XFAKT, VIEWPT(4), TEXT(I))
CALL GEPLT
END

```

Beispiel 1.

```

PROGRAM PIES
*----- Tortendiagramm
REAL      XO /20./, YO /15./, RADIUS /5./,
1         WINKEL /0./, LAENGE /1./, HOEHE /1./
CHARACTER*7 TEXT (5) /'CDU','SPD','FDP','Grüne','sonst'/
REAL      PROZNT (5) / 42.3,      46.3, 5.2, 4.5,      1.7/
INTEGER    SCHRAF (5) / 3033,      4041,3062, 3014,      3048/,
1         FARBE (5) /      1,      2,      3,      1,      2/
COMMON    /GKSCOM/ GKSCOM (6000)
CALL GSPLIT (6000)
CALL GSTXFP (4,2)
CALL DEFPIE (WINKEL, LAENGE, HOEHE, SCHRAF, FARBE, 5)
CALL GENPIE ('O', XO, YO, RADIUS, PROZNT, TEXT, 5)
CALL GEPLIT
END

```

Beispiel 2.

Zur Erzeugung von Kreissektordiagrammen („Tortendiagrammen“) stehen die folgenden Unterprogramme zur Verfügung:

CALL GENPIE (TXTPOS, XO, YO, RADIUS,
WERTE, TEXTE, N)

erzeugt eine Torten-Graphik, die durch folgende Parameter beschrieben wird:

TXTPOS (CHARACTER*(*)) Die Position der Beschriftung der einzelnen Kreissektoren des Diagramms (Abkürzung bis auf einen Buchstaben ist zulässig): 'OUTSIDE' bedeutet, daß die Beschriftung außerhalb der Kreissektoren mit einem kurzen Strich und in der Schriftgröße RADIUS/10 angebracht wird, 'ALIGNED' bringt an jedem der kurzen Mittelstriche zusätzlich einen langen horizontalen Strich und daran erst den Beschriftungstext an, so daß die Beschriftungstexte links und rechts neben dem Diagramm bündig untereinander stehen. 'INSIDE' bringt den Text innerhalb des Sektors in Richtung der Winkelhalbierenden des Sektors an.

XO (REAL) und

YO (REAL) geben den Mittelpunkt des Kreissektordiagramms in Weltkoordinaten an.

RADIUS (REAL) ist der Radius in normierten Gerätekoordinaten (NDC).

WERTE (REAL) ist der Vektor der darzustellenden Werte, die alle größer oder gleich Null sein müssen. Die Werte werden proportional zum

jeweiligen Öffnungswinkel der Sektoren dargestellt.

TEXTE (CHARACTER*(*)) ist ein Textvektor zur Beschriftung der Sektoren.

N (INTEGER) ist die Dimension der letztgenannten Vektoren.

Der Startwinkel der Kreissektoren bezüglich der positiven x-Achse ist implizit auf Null gesetzt. Für die Höhe des Textes und den kurzen Strich vom Kreissektor zum jeweiligen Text wird implizit ein Zehntel des angegebenen Radius verwendet. Auch für den Farb- und Schraffurvektor gelten entsprechend dem Balkendiagramm Voreinstellungen. Wenn diese nicht zusagen, der kann sich des folgenden Unterprogramms bedienen:

CALL DEFPIE (SANGLE, TIKLEN, TXHIGH,
SCHRAF, FARBE, N)

Die Parameter bedeuten:

SANGLE (REAL) Startwinkel des ersten Kreissektors bzgl. der positiven x-Achse (Dieser Sektor und die folgenden Sektoren werden von hier ab entgegen dem Uhrzeigersinn gezeichnet.)

TIKLEN (REAL) ist die Länge des kurzen Striches zum Text bei 'OUTSIDE' bzw. bei 'ALIGNED' der Abstand der Kreisperipherie von der gedachten Vertikalen, an der die Texte auszurichten sind. Dieser Wert sollte stets positiv sein. Er hat keine Bedeutung bei 'INSIDE'.

TXHIGH (REAL) ist die Texthöhe der Beschriftungen.

SCHRAF (INTEGER) Schraffurvektor (vgl. Setzungen bei GSFASI),

FARBE (INTEGER) Farbindexvektor (vgl. Setzungen bei GSFASI),

N (INTEGER) Dimension der beiden letztgenannten Vektoren.

Ein entsprechendes Programm findet sich in Beispiel 2. Bilder für beide Beispiele finden Sie im oben erwähnten Artikel.

GVT – ein neues Terminalprogramm für IBM-PCs

von

D. Lammers

Protokollkonverter im RZ

Ein Protokollkonverter (PK) emuliert das Bildschirmprotokoll einer IBM-Eingabestation (wie sie im Rechenzentrum leider nicht genug herumstehen) auf einem anderen Eingabegerät – üblicherweise einem ASCII-Terminal. Der Benutzer am ASCII-Terminal hat also den Eindruck, an einer IBM-Eingabestation etwa vom Typ 3278 zu sitzen, und auch der Großrechner meint, auf eine IBM-Station zu schreiben.

Nunmehr seit einigen Jahren ist am Rechenzentrum ein Protokollkonverter vom Typ INFOTRON 36X im Einsatz. Speziell für diesen Protokollkonverter wurden auch die Programme MT (Micro Terminal) und GMT (Graphic Micro Terminal) geschrieben, die es ermöglichen, einen IBM-PC als Terminal einzusetzen und sogar Dateien zu übertragen bzw. graphische Ausgaben zu erhalten. Leider ist der INFOTRON-PK sehr empfindlich, so daß Überlegungen angestellt wurden, einen neuen Protokollkonverter vom Typ IBM 3708 anzuschaffen und in Betrieb zu nehmen. Dieser hat mehrere sehr vielversprechende Fähigkeiten und sollte auch wesentlich stabiler als der alte sein. Der IBM 3708 versteht sich aber leider nicht ideal mit MT/GMT, und aus mehreren Gründen empfahl es sich nun, gleich ein neues Terminalprogramm zu schreiben, in dem Datei-Transfer und Graphikfähigkeiten zusammengefaßt werden konnten. Dieses neue Programm, GVT (für Graphic Video Terminal) ist nun seit einiger Zeit im Testeinsatz und kann mit

Erscheinen des **inforum** allen Interessenten zur Verfügung gestellt werden.

Das neue Programm GVT

(So ganz „neu“ ist GVT auch nicht, der Kern des Programms beruht auf einem von Stephan Heuer und mir entwickelten Terminalprogramm, das seit einiger Zeit im FB 15 und an anderen Stellen im Einsatz ist.)

GVT emuliert auf einem IBM-PC eine Mischung aus VT100/VT131-Terminal (eingetragenes Warenzeichen der Fa. DEC). Insofern unterscheidet es sich nicht wesentlich von MT/GMT. Es hat aber noch einige andere Fähigkeiten, die im folgenden einzeln aufgeführt sind:

Jede Taste des IBM-PCs läßt sich mit einem String belegen (der auch Sonderzeichen enthalten kann). Das kann sowohl interaktiv (während das Programm läuft) als auch beim Starten aus einer oder mehreren DOS-Dateien erfolgen. In einer solchen Tastatur-Datei können auch Leitungsparameter festgehalten werden. Beim Aufruf von GVT liest das Programm automatisch eine Datei „SETUP.VT“, die die üblichen Leitungsparameter (Baud-Rate, Parität, Anzahl der Stopbits) und die normalen Tastaturbelegungen enthält, so daß das umständliche Einstellen der Leitungsparameter von Hand entfällt.

Bei genügend großem Hauptspeicher kann interaktiv (und in Zukunft auch von CMS aus) auf DOS zugegriffen werden, d.h. man kann DOS-Kommandos ausführen, ohne GVT zu verlassen.

GVT speichert intern 8 Bildschirmseiten. Seite 0 ist die DOS-Seite, die beim Ausführen von DOS-Kommandos und nach dem Verlassen des Programmes angezeigt wird. Seite 1 ist die übliche Arbeitsseite. Die Seiten 2 bis 7 stehen dem Benutzer zur Verfügung, er kann hier z.B. ein File-Listing stehen lassen, um nicht jedesmal neu „fl“ aufrufen zu müssen.

Der File-Transfer ist tatsächlich ganz neu: Er wird vollständig von CMS aus gesteuert (d.h. ein CMS-Kommando wird aufgerufen, und der Transfer beginnt). Es ist möglich, mehrere Dateien auf einmal zu transferieren, außerdem kann man natürlich zwischen Text- und Binärdateien wählen, ebenso kann man wählen, ob von ASCII nach EBCDIC bzw. umgekehrt übersetzt werden soll oder nicht, und man kann sogar eine eigene Übersetzungsta-

belle angeben, die die Standard-Tabelle überschreibt.

In den schon oben erwähnten Tastatur-Dateien bzw. in „SETUP.VT“ kann man 7 besondere Strings angeben. Es handelt sich dabei um

1. einen String, der beim Starten des Programms an CMS gesendet wird,
2. einen String, der beim Verlassen des Programms an CMS gesendet wird,
3. einen String, der im Normalzustand in der Statuszeile angezeigt wird,
4. einen String, der bei gedrückter Shift-Taste in der Statuszeile angezeigt wird,
5. einen String, der bei gedrückter Control-Taste in der Statuszeile angezeigt wird,
6. einen String, der bei gedrückter Alternate-Taste in der Statuszeile angezeigt wird,
7. einen String, der bei gedrückter Alternate- und Control-Taste in der Statuszeile angezeigt wird.

Wie schon GMT kann mit GVT wie mit einer normalen Graphik-Workstation des RZ verfahren werden: man kann sich mit PPV oder interaktiver GKS-Programmierung Bilder anschauen, kann mit einem geeigneten Hardcopy-Gerät (z.B. IBM Graphic-Printer) Hardcopies machen und die Bilder auf Diskette sichern, um sie später noch einmal zu betrachten oder auszudrucken.

Wie schon bei MT ist die Geschwindigkeit des File-Transfers vom PC zum CMS sehr langsam. Etwas Abhilfe schafft da vielleicht das neue VM-Release. Es soll aber nicht unerwähnt bleiben, daß der Transfer im Vergleich zu anderen Mög-

lichkeiten (Ethernet, ...) immer relativ langsam sein wird. In der anderen Richtung ist eine ganz akzeptable Transferrate von 100 Bytes pro Sekunde erreichbar. Für die Graphikfähigkeiten ist es immer eine interessante Frage, welche Geräte bzw. Graphikkarten jetzt von uns unterstützt werden. Bislang wird die weit verbreitete Hercules-Karte sowie das Tecmar-Graphic-Board und die Tektronix-Geräte der Serie 401X als graphische Monitore sowie Drucker der Typen Epson FX80, NEC P2, IBM Graphic Printer 1 unterstützt. Treiber für die EGA-Karte und die NEC-Drucker P3 und P6/P7 sind in Vorbereitung.

Schlußbemerkungen

Wie immer ist die Dokumentation nicht rechtzeitig fertig, ich hoffe aber, bis zum Erscheinen des Artikels eine lesbare Dokumentation und ein Help-Menü fertig zu bekommen.

Anders ist das Vorgehen beim Einloggen und Ausloggen. Hierauf wird aber in der Dokumentation noch eingegangen werden.

Es sind im Laufe der Zeit mehr Graphiktreiber zu erwarten, die eventuell auch Farbausgabe ermöglichen. (Die EGA-Karte wäre die erste dieser Reihe.)

Auch in der Geschwindigkeit des Filetransfers wird sich noch einiges tun, im Moment ist dafür aber noch kein Termin in Sicht.

Ich hoffe, die Benutzer werden mit GVT soweit zufrieden sein – bislang gab es von den Testbenutzern Lob, aber auch (kreative) Kritik. Alle Anmerkungen zu GVT bitte an mich: URZ65 at DMSWWU1A.

RUM-Tutorial

Bildschirmsteuerung im Dialog mit ISPF

von

A. Achilles

In den letzten beiden Ausgaben des **inforuM** wurde die Benutzung von Variablen besprochen. Dabei wurde eine Kategorie von Variablen, die Systemvariablen, besonders vorgestellt. Diese Variablen dienen dazu, spezielle Informationen zwischen den Anwendungen und dem steuernden System ISPF auszutauschen. Der Beitrag in dieser Ausgabe soll noch einige Systemvariablen vorstellen, die besonders nützlich für die Gestaltung umfangreicher Anwendungen sind – die Steuerung mit Hilfe der Funktionstasten wird dadurch möglich –, ferner sollen Steuervariablen besprochen werden, die dazu dienen, in einer Maske bestimmte Funktionen auszulösen bzw. Bedingungen zu testen.

Systemvariablen für die Funktionstasten

Die Setzung der Funktionstasten wird in den Systemvariablen ZPF01, ... , ZPF24 gespeichert. Diese Variablen sind Ein-/Ausgabevariablen, die im Anwendungs-Profil gespeichert werden. Pro Variable können bis zu 255 Zeichen eingegeben werden. An einem Bildschirm mit 24 Funktionstasten ist die Zuordnung so, wie man es vermuten würde: der Funktionstaste 1 ist der Wert der Variablen ZPF01 zugeordnet, ... , der Taste 24 der Wert von ZPF24. Arbeiten Sie jedoch an einem Bildschirmgerät, das nur 12 Funktionstasten besitzt, so werden Sie erleben, daß der Funktionstaste 1 der Wert der Variablen ZPF13, ... , der Taste 12 der Wert der Variablen ZPF24 zugeordnet ist.

Wird eine Maske auf dem Bildschirm präsentiert und drücken Sie auf eine der Funktionstasten, so enthält die Steuervariable PFKEY anschließend den Namen der gedrückten Funktionstaste, also „PF01“, ... , „PF24“. Ist in der Maske eine Vereinbarung für die Variable ZCMD vorgenommen worden, d.h. können in der Maske auch Kommandos eingegeben werden, so geschieht noch mehr: der Wert der zugehörigen Variable ZPFxx wird in der Variablen ZCMD gespeichert und die Verarbeitung erfolgt so, als ob Sie in die Kommandozeile das entsprechende Kommando eingetragen hätten.

Wie jede Regel, so hat auch diese ihre Ausnahmen:

- Beginnt der Wert, der der Variablen ZPFxx zugewiesen wurde, mit einem Doppelpunkt (:), so wird der Doppelpunkt entfernt und der übrige Wert der ersten Variablen in derjenigen Zeile zugewiesen, in der der Cursor gerade steht.
- Beginnt der Wert der Variablen ZPFxx mit dem Zeichen >, so wird das Kommando an die Variable ZCMD übertragen und direkt ausgeführt.

Steuervariablen

In der Variablen PFKEY ist Ihnen die erste Steuervariable begegnet. Eingangs wurde darauf hingewiesen, daß Steuervariablen dazu dienen, Funktionen in den Masken auszulösen bzw. Bedingungen zu überprüfen. Aus diesem Grunde können Sie Steuervariablen auch nur in Masken einsetzen, und zwar in den Abschnitten, die der Initialisierung oder Bearbeitung dienen:

```
INIT
REINIT
PROC
```

Steuervariablen unterscheiden sich von den übrigen Dialogvariablen auch noch in der Weise, wie sie in einer Maske aufgerufen werden. Alle Steuervariablen beginnen dort mit einem Punkt (.). Die Steuervariable PFKEY wird somit in der Maske mit

```
.PFKEY
```

angesprochen.

Nun sollen die wichtigsten Steuervariablen aufgelistet und ihre Eigenschaften erläutert werden.

ALARM

Diese Variable dient dazu, beim Anzeigen einer Maske den Alarm auszulösen bzw. die Einstellung, ob der Alarm ausgelöst wird, anzuzeigen. Als Werte sind „YES“, „NO“ bzw. ein Leerzeichen zuge-

lassen (NO und Leerzeichen sind identisch). Wird in der Wert dieser Steuervariablen auf YES gesetzt, so wird der Alarm ausgelöst, andernfalls wird kein Alarm gegeben – es sei denn, es wird eine Meldung angezeigt (siehe unten), in der ein Alarm ausgelöst wird. Die Steuervariable kann auch dazu benutzt werden, den derzeitigen Wert in einer Dialogvariablen zwischenzuspeichern.

Im folgenden Beispiel wird der derzeitige Wert der Dialogvariablen WERT benutzt, um das Auslösen eines Alarms zu steuern; sodann wird in der Variablen NEUWERT gespeichert, ob ein Alarm ausgelöst wurde.

```
.ALARM = &WERT
&NEUWERT = .ALARM
```

ATTR

Mit Hilfe dieser Steuervariablen können Sie die Feldattribute verändern. Die Syntax lautet

```
.ATTR (feld) = 'schlüssel (wert) ...'
```

Dabei kann feld folgende Werte annehmen:

- den Namen eines Ein- bzw. Ausgabefeldes in der Maskendefinition,
- den Wert „.CURSOR“ – damit ist dasjenige Feld gemeint, das der Steuervariablen CURSOR zugeordnet ist – oder
- eine Dialogvariable, die den Namen eines Ein- bzw. Ausgabefeldes in der Maskendefinition enthält.

Die Kombination schlüssel(wert) muß ein gültiges Attributschlüsselwort und einen dazu passenden Wert enthalten, die in entsprechender Form auch im ATTR-Abschnitt der Maskendefinition benutzt werden dürfen. Bei der Verwendung der Steuervariablen ATTR können mehrere Attribute zugleich geändert werden. Dabei ist jedoch darauf zu achten, daß beim TYPE-Attribut nur die Werte „INPUT – OUTPUT“ bzw. „DATAIN – DATAOUT“ ineinander überführt werden können.

Wird die Steuervariable im Abschnitt INIT benutzt, so gilt die Definition für das Feld solange, bis diese Definition erneut überschrieben wird, wird ATTR in den Abschnitten REINIT bzw. PROC benutzt, so gilt die Veränderung nur für das nächste Anzeigen der Maske.

Das folgende Beispiel läßt den im Feld PASSWORT eingetragenen Wert sichtbar werden und das Feld, das der Steuervariablen CURSOR im Moment zugeordnet ist, blinken und ändert es in ein Eingabefeld (dabei wird vorausgesetzt, daß es vorher ein Ein- bzw. Ausgabefeld war):

```
.ATTR (PASSWORT) = 'INTENS (LOW)'
.ATTR (.CURSOR) = 'TYPE (INPUT)
HILITE(BLINK)'
```

CURSOR

Wird diese Variable im INIT- bzw. REINIT-Abschnitt gesetzt, so wird damit die Stellung des Cursors zu Beginn der Maskendarstellung beeinflusst. Der Wert muß dabei eine Zeichenkette sein, die einem gültigen Feldnamen entspricht. Der Cursor wird dann zu Beginn auf das angegebene Feld positioniert.

Diese Variable wird automatisch auf das zuletzt angesprochene Feld gesetzt, wenn die Steuervariable MSG entweder direkt gesetzt oder indirekt durch die Funktionen TRANS bzw. VER verändert wird. Im unten stehenden Beispiel wird der Cursor auf das Feld EINGABE positioniert und dieser Wert in der Variablen &CNAME zwischengespeichert:

```
.CURSOR = EINGABE
&CNAME = .CURSOR
```

HELP

Diese Variable kann im INIT-Abschnitt benutzt werden und dient dazu, bei Aufruf des HELP-Kommandos die Maske auf dem Bildschirm darzustellen, deren Name in der Variablen HELP vereinbart wurde:

```
.HELP = HILFE1
```

PFKEY

Wie bereits oben besprochen, dient diese Steuervariable dazu, nachzuhalten, welche Funktionstaste Sie betätigt haben. Wurde keine Funktionstaste gedrückt, so enthält PFKEY Leerzeichen, andernfalls die Zeichenkette PFxx, wobei xx für die Nummer der gedrückten Taste steht. Die Variable kann nur im PROC-Abschnitt verwendet werden. Im nachfolgenden Beispiel wird eine Meldung erzeugt, wenn die Funktionstaste 5 gedrückt wurde, und in der Variablen TASTE wird gespeichert, ob und ggf. welche Funktionstaste gedrückt wurde.

```
IF (.PFKEY = PF05)
  .MSG = PAN01
&TASTE = .PFKEY
```

MSG

Diese Variable kann die Nummer einer Meldung zugewiesen bekommen. Dies wird üblicherweise im PROC-Abschnitt gemacht und bewirkt, daß die Meldung, die zu der angegebenen Nummer gehört, auf dem Bildschirm erscheint.

RESP

Die Steuervariable enthält die Werte „ENTER“ oder „END“, je nachdem, ob Sie eine Maske „normal“ oder durch den „END“- bzw. „RETURN“-Befehl (in der Regel die Funktionstasten PF3 bzw. PF4) verlassen haben.

Wird dieser Variablen im INIT- bzw. REINIT-Abschnitt der Wert „END“ oder im Abschnitt PROC der Wert „ENTER“ zugewiesen, so wird damit beeinflußt, ob die Maske am Bildschirm dargestellt wird oder ob sie mit dem END bzw. RETURN-Kommando verlassen werden kann.

TRAIL

Die Funktion TRUNC (siehe *inforum* Nr.1/1987) liefert den Anfang des Wertes einer Variablen. Der Rest wird der Steuervariablen TRAIL zugewiesen. Wird zur Trennung ein spezielles Zeichen herangezogen, z.B. der Punkt (.), so enthält TRAIL die diesem Zeichen folgende Zeichenkette.

ZVARS

Dieser Variablen kann im Initialisierungsabschnitt eine Zeichenkette mit den Namen von Ein-/Ausgabefeldern der Maske zugewiesen werden. Im Kör-

per der Maske stehen dann für die Feldnamen jeweils der Platzhalter Z. Die Zuordnung wird bestimmt durch die Reihenfolge der Platzhalter Z in der Maskendefinition und die Reihenfolge der Namen in der Zuweisung zur Variablen ZVAR. Diese Technik kann verwendet werden, um Ein-/Ausgabefelder in der Maske zu benutzen, deren Namen länger sind als die zur Darstellung verwendeten Zeichen.

Ausblick

Das in den vier Beiträgen vorgestellte Werkzeug ISPF enthält noch eine Reihe von weiteren Eigenschaften, die hier nicht vorgestellt wurden. Sicher werden Sie auch einiges an Komfort vermissen – ISPF ist zwar ein vielseitiges, an manchen Stellen aber auch unflexibles Produkt, das zudem in der Initialisierungsphase eines Dialoges auf Seiten des Systems einen hohen Aufwand erfordert. Trotzdem scheint es mir beim Aufbau größerer Dialog-Anwendungen sinnvoll zu sein, auf ein Werkzeug wie ISPF zurückzugreifen. Der Aufwand, den das System in der Initialisierungsphase treiben muß, sollte Sie andererseits davon abhalten, eine kleine Dialoganwendung – z. B. mit nur einem Aufruf einer Bildschirmmaske – mit ISPF zu gestalten.

Nicht erwähnt wurde in dieser Serie z.B. die Möglichkeit, (kleine) Tabellen in ISPF zu verwalten, ISPF auch für den Batch-Betrieb zu benutzen, usw..

Diese Serie sollte an einigen Stellen ausführlich die Eigenschaften darstellen, in anderen Punkten nur Richtungen aufzeigen. Vielleicht müssen Sie demnächst eine größere Dialog-Anwendung erstellen und erinnern sich dann an die Möglichkeiten, die ISPF Ihnen bietet.

Wenn Sie bei Fragen zu ISPF Beratung brauchen, wenden Sie sich bitte an mich (URZ06, Tel. 2607).

RUM-Aroma

inforum-Quiz

von

E. Sturm

Zunächst die Auflösung des letzten Quiz: Nach Durchlaufen des Programmabschnitts

```
DECLARE A (8,8) BIN FIXED
INIT ((21)(1,2,3),1);
A = (A < 3) * (A + 1) + (A = 3);
```

verändert sich A, als ob es folgendermaßen deklariert wäre:

```
DECLARE A (8,8) BIN FIXED
INIT ((21)(2,3,1),2);
```

d.h. alle Elemente werden um 1 erhöht, der höchste Wert wieder auf 1 gesetzt.

Dies kann man verstehen, wenn man sich vor Augen hält, daß eine Vergleichsoperation in PL/I immer einen BIT-Wert liefert, der Vergleich zweier Arrays also einen BIT-Array! $(A + 1)$ nun ist der um 1 erhöhte Array A. Multipliziert man diesen mit dem BIT-Array $(A < 3)$, so werden alle Elemente von $(A + 1)$ mit 1 oder 0 multipliziert, je nachdem, ob das entsprechende Element von A kleiner als 3 war oder nicht. Man erhält also

$A(i) + 1$	für $A(i) < 3$ und
0	sonst.

Der BIT-Array $(A = 3)$ ist

'1'B	für $A(i) = 3$ und
'0'B	sonst.

Da nur die Werte 1 bis 3 vorkommen, ist also die Summe

$A(i) + 1$	für $A(i) < 3$ und
1	für $A(i) = 3$.

Zu erwähnen bleibt noch, daß BIT-Werte zum Rechnen automatisch nach BIN FIXED umgewandelt werden.

Nun das neue Problem:

In vielen PL/I-Klausuren findet man die Aufgabe, analog zur VERIFY-Builtin-Funktion eine FALSIFY-Funktion zu schreiben, die als Wert die erste Position in STRING1 zurückgibt, an der ein Zeichen aus STRING2 steht:

FALSIFY ('HONOLULU', 'AEIOU')

liefert also 2, da der erste Vokal in HONOLULU an zweiter Stelle steht. Fälschlicherweise wurde dabei (auch von mir) meist noch erwähnt, daß dies nicht mit Builtin-Funktionen möglich sei. Suchen Sie die beiden Builtin-Funktionen X und Y, mit deren Hilfe Sie durch Schachtelung die Wirkung von FALSIFY erzielen können. Falls Sie nicht darauf kommen, es sieht ungefähr so aus:

$X(Y('HONOLULU', ?), 'AEIOU'), ?)$

Auflösungen bitte an URZ31 at DMSWWU1A.

Känzeln Sie Ihren Auswurf?

von

A. Benduhn-Mertz

Sicher haben Sie schon bewundert, wie schön die Rechenzentrumskollegen die deutsche Sprache beherrschen. Es gibt doch keine Mitteilungen in den „NEWS“ oder auch keine Artikel im *inforum*, die in der Art und Weise, wie ich es in der Überschrift tat, unsere Muttersprache vergewaltigen. Sollte Ihnen dieser Umstand bisher nicht aufgefallen sein, dann liegt das sicher daran, daß Sie nichts anderes als normales Deutsch erwarten und bisher nicht enttäuscht wurden.

Aber haben Sie auch schon mit uns gesprochen, oder gar einem Gespräch zwischen zwei Rechenzentrümlern gelauscht? Da hört's dann mit dem Gebrauch der Muttersprache ganz schnell auf, und Sie lauschen schönstem Babylonisch.

Zumindest in den Vorlesungen möchte ich natürlich, wie die Bayern sagen würden, „schriftdeutsch“ reden und den fremden Zungenschlag umgehen. Eine Diskette darf dabei aber nicht zum Plättlein geraten (und keine kleine Datei zum Filechen). Hier gelingt vielleicht noch die Umschreibung oder gute Übersetzung.

Schwieriger wird es da, wo die Kollegen in ihren Artikeln zu optischen Auszeichnungen greifen. Da mir im gesprochenen Wort Anführungsstriche, spitze Klammern und Kursivdruck fehlen, weiß ich dann nicht, wie ich den „JOB“ sprachlich bewälti-

gen soll. An dieser Stelle wird es zusätzlich semantisch schlimm. Erledigt der Rechner eigentlich seine oder meine Aufgabe, oder ist es meine Aufgabe, ihn arbeiten zu lassen? Dann arbeite ich also gar nicht mehr selbst? Oh doch, denn da bleibt ja noch das „DEBUGGING“. Wer wagt es schon, von Entkäfern oder Entwanzen zu reden. Fehler-suche ist ja weniger.

Beim „OUTPUT“ gerate ich dann endgültig ins Stottern, der Auswurf klingt mir zu medizinisch und die Ausgabe ist auch schon anders belegt.

Vielleicht muß ich doch einmal einen niedergelassenen Germanisten konsultieren und mir einschlägige, möglichst elegante Übersetzungsangebote

unterbreiten lassen. Oder ich werde der Dudenredaktion vorschlagen, Wörter wie känzeln und purgen einzudeutschen und in das heilige Werk mit aufzunehmen. Vielleicht kann ich dann auch gleich den Vorschlag machen, kompatibel und optimal endlich als steigerungsfähig anzuerkennen, das Wort ‚editieren‘ aufzunehmen (schließlich denkt ja längst nicht jeder Benutzer, der mit einem Editor, also einem ‚Herausgeber‘ arbeitet, daran, seine Schöpfung dann auch wirklich an die Öffentlichkeit zu bringen) und als wirklich letztes, das Wort ‚physisch‘ zu streichen und die Bedeutung von ‚physikalisch‘ zu erweitern, oder denken Sie darüber nach, in welcher ‚physischen‘ Form Sie diesen Artikel vorliegen haben?