

inforum

INforMationsforum des Rechenzentrums der Universität Münster

Nr. 2

April 1977

Inhalt

Umstellung auf den PL/I Optimizing Compiler	1
NAG-Bibliothek	2
Informationen über den Rechenbetrieb	3
Der PL/I Optimizing Compiler	5
Zum Einsatz der Batch-Compiler	9
PL/C Release 7.5	21
Drucken und Stanzen in ALGOL W	21
FORTTRAN-Programmeditierung	22
Lehrangebot des Rechenzentrums	23
Leserforum	26
Programmquerschnitt März 1977	28

Umstellung auf den PL/I Optimizing Compiler

H. Meyer

Es ist beabsichtigt, den PL/I(F)-Sprachübersetzer am Ende des Jahres aus dem Betriebssystem herauszunehmen, da eine ordnungsgemäße Wartung dieses Übersetzers nicht mehr erfolgen kann. Wir empfehlen daher, die Umstellung aller Produktionsprogramme auf den PL/I Optimizing Compiler (PL0-Prozeduren) im Verlaufe des nächsten halben Jahres vorzunehmen und beim Auftreten von Schwierigkeiten diese dem Rechenzentrum umgehend mitzuteilen. Zur Erleichterung der Umstellung können die Broschüren zum PL/I Optimizing Compiler einmalig über das Rechenzentrum bezogen werden.

IBM-Form No. GC33-0009-4 : PL/I Language Reference Manual

IBM-Form No. GC33-0006-4 : PL/I Optimizing Compiler:
Programmer's Guide

Preis je Broschüre: ca. 35 DM

Bestellung bis zum 31.5.77 bei Frau E. Geisler, Zi.: 101,
Tel: 2460.

NAG-Bibliothek

U. Ebert

Für die Behandlung numerischer Probleme steht jetzt auch die NAG-Bibliothek zur Verfügung (NAG = Numerical Algorithm Group). Es handelt sich hierbei um eine Sammlung von hochwertigen, solide getesteten und gut dokumentierten FORTRAN-IV-Unterprogrammen. Die jetzige Version "Mark 5" wurde seit 1970 durch die Zusammenarbeit einer Reihe englischer Universitäten entwickelt. Sie ist im Umfang mit dem vorhandenen SSP (Scientific Subroutine Package) vergleichbar und wird in absehbarer Zeit das SSP ersetzen.

Die Vorteile der NAG-Bibliothek liegen darin, daß die Programme von Numerikern entwickelt wurden, und dabei auch neuere Erkenntnisse der numerischen Mathematik sowie die Eigenschaften des Maschinentyps IBM/360 berücksichtigt wurden. Außerdem sind diese Programme unabhängig vom Autor noch jeweils von einem anderen Mitglied der Gruppe einem harten Test unterzogen worden. Das hat eine hohe Zuverlässigkeit und gute Präzision der Ergebnisse zur Folge. Darüberhinaus wird die Anwendung durch die unten beschriebene Dokumentation sehr leicht gemacht.

Die vorhandene Version benutzt doppelte Genauigkeit (REAL*8 - Größen).

Dokumentation und Beratung bei der Benutzung:

Bei der Benutzung der NAG-Bibliothek helfen folgende Unterlagen:

a) NAG Mini Manual

Es gibt einen Überblick über die vorhandenen Programme, den mathematischen Hintergrund des betreffenden Gebiets und Empfehlungen für die Auswahl eines speziellen Programms. Es sollte zuerst konsultiert werden.

b) NAG Library Manual (Band 1 - 3)

Hier sind alle Programme genau dokumentiert. Unter anderem wird dort für jedes Programm der Algorithmus beschrieben, Literatur angegeben, und insbesondere werden die Parameter erklärt. Ein Beispielprogramm demonstriert den Aufruf und die Ergebnisse.

Die Manuals liegen in der Programmierberatung (Zimmer 06, Roxeler Str. 60) und in der Bibliothek des Rechenzentrums (Zimmer 23, Hittorfstr. 27) zur Einsicht aus.

Interessenten können sich an einer Sammelbestellung beteiligen (Anmeldung dazu bis zum 31.5.77 bei Frau E. Geisler, Zi.: 101, Tel.: 2460; Preis: ca. 25 DM für das Mini Manual, ca. 80 DM für die Library Manuals).

Zusätzlich bietet das Rechenzentrum eine Beratung zur Benutzung der NAG-Bibliothek an:

Mo-Fr 11-12, Zimmer 108, Roxeler Str. 60.

Sie steht bei allen Fragen und Schwierigkeiten mit der NAG-Bibliothek und der sonstigen Programmbibliothek zur Verfügung.

Steuerkarten für den Aufruf:

Die NAG-Bibliothek ist mit den normalen FORTRAN-Prozeduren benutzbar. Es muß lediglich ein zusätzlicher symbolischer Parameter angegeben werden:

```
//name EXEC FORTCE,SUBLIB=NAG
//COMPILE.SYSIN DD *
    Programme mit Aufruf von
    NAG-Unterprogrammen
//EXECUTE.SYSIN DD *
    Daten
//
```

In diesem Fall ist die NAG-Bibliothek verfügbar und nicht das SSP. Läßt man SUBLIB=NAG weg, so ist nur das SSP verfügbar.

Wie andere FORTRAN-Programme sind die NAG-Programme ebenfalls von mit dem PL/I-Optimizer übersetzten PL/I-Programmen und von ALGOL-W-Programmen aus aufrufbar. In diesen Fällen müssen zusätzliche Steuerkarten zur jeweiligen Standardprozedur hinzugefügt werden.

Informationen über den Rechenbetrieb

K. Reichel

Angeregt durch einen Verbesserungsvorschlag eines inforum-Lesers sollen an dieser Stelle einige Informationen über den derzeitigen Rechenbetrieb gegeben werden. Zunächst wird dargestellt, nach welchen Kriterien die Operateure die einzelnen Job-Warteschlangen abarbeiten. Innerhalb der Warteschlangen werden die Jobs entsprechend ihrer Priorität bearbeitet.

Zu folgenden Betriebszeiten haben die verschiedenen Job-Typen bzw. Job-Klassen Vorrang:

- Montag bis Freitag, 7.30 -- 20.00 Uhr
Jobs mit Zeitangaben bis zu 4 Minuten (Testläufe: MINI, TEST oder NORM) in der Reihenfolge der Klassen E, A, G UND K. Batch-Läufe werden zu jeder vollen Stunde abgearbeitet.
- Montag bis Freitag, 20.00 -- 22.00 Uhr
In dieser Zeit sollen bevorzugt die Testläufe abgearbeitet werden, die bis 20.00 Uhr eingelesen wurden.
- Montag bis Donnerstag, 22.00 -- 7.30 Uhr
Jobs mit Zeitangaben über 4 Minuten entsprechend ihrer Priorität.
- ab Freitag, 22.00 Uhr
In dieser Zeit soll bevorzugt der Überhang der vergangenen Woche abgearbeitet werden.

Dec-PL/I-Optimizing-Compiler

H. Meyer

Dies ist eine Fortsetzung der in inform Nr. 1 begonnenen Beschreibung; Einzelheiten können der oben angegebenen Literatur entnommen werden.

3. Neue Anweisungen und Spracherweiterungen**- DEFAULT-Anweisung (Abkürzung: DFT)**

Die Default-Anweisung gestattet die Angabe von nicht ausdrücklich aufgeführten Attributen für

1. explizit erklärte Identifizier,
2. durch Kontext erklärte Identifizier,
3. implizit erklärte Identifizier,
4. Parameterbeschreibungen (im ENTRY-Attribut) und
5. die Funktionswerte interner Prozeduren.

Bemerkung: Die Benutzung der DEFAULT-Anweisung sollte in keinem Fall die explizite Vereinbarung von Variablen ersetzen, sie gestattet es jedoch, auf einfache Weise gewisse Attribute (wie die Genauigkeit von Zahlen oder die Länge von Strings) in größeren Programmabschnitten zu kontrollieren.

- SELECT-Anweisung

Die SELECT-Anweisung definiert den Anfang einer Auswahl-Gruppe; diese ermöglicht die Auswahl einer Anweisungsfolge aus einer größeren Zahl solcher Anweisungsfolgen durch Angabe von Bedingungen ähnlich einer IF-Anweisung, wobei jedoch nicht nur zwei Möglichkeiten zur Auswahl zu stehen brauchen. Die Auswahl-Gruppe wird durch eine END-Anweisung abgeschlossen.

Beispiel:

```
SELECT;
    WHEN (X>10) DO; ... END;
    WHEN (X>=0) CALL TEST(X);
    OTHERWISE   CALL FEHLER;
END;
```

Aufbau:

```
SELECT [(expr-1)];
    [[(condition-prefix [,condition-prefix]...):]
    WHEN (expr-2 [,expr-2]...) unit]...
    [{OTHERWISE} unit]
    { OTHER
    }
END [identifizier];
```

mit expr-1 und expr-2 derart, daß ((expr-1) = (expr-2)) einen skalaren Wert vom Typ BIT besitzt,

und mit unit:

 einfache Anweisung

 (außer DO, SELECT, PROCEDURE, BEGIN, END,
 DECLARE, DEFAULT, FORMAT, ENTRY)

 DO-Gruppe

 Auswahl-Gruppe

 BEGIN-Block

- Leave-Anweisung

Die LEAVE-Anweisung bewirkt den normalen Abschluß einer oder mehrerer geschachtelter DO-Gruppen. Falls keine LABEL-Konstante dem LEAVE folgt, muß sie im Innern einer nach der Ausführung zu verlassenden DO-Gruppe stehen. Falls eine LABEL-Konstante in der LEAVE-Anweisung angegeben ist, muß eine DO-Anweisung mit diesem LABEL existieren, deren Gruppe die LEAVE-Anweisung umfaßt und die nach ihrer Ausführung verlassen werden soll.

Beispiel:

```
LX: DO I=1 TO 10;
      DO J=1 TO 10;
        IF X(I,J)=0 THEN LEAVE LX;
      ;
    END;
  ;
END LX;
```

LL:

(Bei Auftreten einer "0" wird bei Label LL weitergearbeitet.)

Aufbau:

LEAVE [label-constant];

- Optionen in der DO-Anweisung und in Wiederholungsangaben in Datenlisten für STREAM-I/O

Mit der UNTIL-Option wird die Wiederholung von Schleifen ähnlich kontrolliert wie mit der WHILE-Option, im Unterschied zu dieser erfolgt der Test auf die Erfüllung einer Bedingung jedoch am Ende der Schleife; insbesondere wird die Schleife mindestens einmal durchlaufen und eine gegebenenfalls vorhandene Kontrollvariable im Fall einer erfüllten Bedingung in der UNTIL-Option nicht mehr verändert.

Mit der REPEAT-Option wird der Kontrollvariablen einer Schleife vor jeder Wiederholung ein neuer Wert zugewiesen; dabei wird der zugeordnete Ausdruck jeweils neu ausgewertet.

Beispiel:

```
DO A=1 REPEAT A*X UNTIL (A<1E-10);
  PUT LIST (A);
END;
```

(Diese Schleife gibt die Anfangsglieder der geometrischen Reihe $1 + x + x^2 + \dots$ aus.)

Aufbau:

DO-Anweisung:

```
DO { WHILE(expr-4) [ UNTIL(expr-5)] } ;
  { UNTIL(expr-5) [ WHILE(expr-4)] }
DO { pseudovariablen } = specification [, specification]...;
  { variable }
```

Wiederholungsangaben in Datenlisten:

(element [, element]...)

```
DO { pseudovariablen } = specification [, specification]...
  { variable }
```

mit specification:

```

expr-1 [ TO expr-2 [ BY expr-3 ] [ WHILE(expr-4) [ UNTIL(expr-5) ]
        BY expr-3 [ TO expr-2 ] [ UNTIL(expr-5) [ WHILE(expr-4) ]
        REPEAT expr-6 ] ] ]
    
```

mit expr-1: element expression

mit pseudovariablen: IMAG, REAL, SUBSTR, UNSPEC, ONCHAR,
ONSOURCE, STATUS

- Neue Datentypen
Die Einführung von ENTRY- und FILE-Variablen ermöglicht es, die zu benutzenden Prozeduren und Files während der Ausführung festzulegen oder auszutauschen (z.B. durch Zuweisung oder Indizierung).
- Neue "BUILT-IN"-Funktionen
ACOS, ASIN
PLIRETV
CURRENTSTORAGE(CSTG), STORAGE(STG)
OFFSET, POINTER
- STRINGSIZE-Bedingung (Abkürzung: STRZ) und Bedingungs-Präfixe
STRINGSIZE(STRZ) - NOSTRINGSIZE(NOSTRZ)
Diese Bedingung ermöglicht die Kontrolle der Zuweisung von Strings; falls die Bedingung durch einen STRINGSIZE-Bedingungs-Präfix aktiviert wurde, wird bei Zuweisungen geprüft, ob hierbei die zugewiesene Zeichenkette abgeschnitten werden muß.
- Neue ENVIRONMENT-Optionen
Die Bandpositionierung nach der Bearbeitung einer Banddatei wird durch LEAVE oder REREAD gesteuert. Die Wirkung von LEAVE ist gegenüber dem PL/I(F)-Übersetzer geändert; eine ENV(LEAVE) oder ENV(REREAD)-Option kann auch in der CLOSE-Anweisung angegeben werden.
Ein-/Ausgabe-Anweisungen für Record-I/O können vom Übersetzer optimiert werden (keine Unterprogrammaufrufe), falls dem Übersetzer alle wichtigen Parameter der Datei und der zu übertragenden Variablen zur Verfügung stehen; dann kann ENV(TOTAL...) angegeben werden.
Für das Arbeiten mit Sätzen variabler Länge kann SCALAVARYING angegeben werden. Die Sätze werden dann mit einer PL/I-spezifischen Längenangabe versehen, die das Verarbeiten dieser Sätze mit PL/I-Programmen vereinfacht, was jedoch bei anderen Programmen berücksichtigt werden muß.

4. Anweisungen zur Steuerung der Programmlisten

Es stehen vier Steueranweisungen zur Verfügung, mit denen die Ausgabelisten des Übersetzers beeinflusst werden können;

diese Anweisungen sollten möglichst anstelle der Vorschubzeichen (in Spalte 1) verwendet werden.

- %PAGE;
bewirkt die Fortsetzung der Übersetzerlisten (INSOURCE wie auch SOURCE) auf der ersten Zeile der folgenden Seite.
- %SKIP [(n)];
gibt die Anzahl der Leerzeilen ($1 \leq n \leq 999$) an, die auf diese Anweisung in der Übersetzerliste (INSOURCE wie auch SOURCE) folgen sollen.
- %NOPRINT;
bewirkt die Unterdrückung der Druckausgabe der folgenden Programmtexte (INSOURCE wie auch SOURCE) durch den Übersetzer.
- %PRINT;
hebt die Wirkung der %NOPRINT-Anweisung wieder auf. Der folgende Quelltext wird aufgelistet.

Die aufgeführten Anweisungen dürfen nicht innerhalb anderer Anweisungen auftreten und sollten auf eigenen Karten stehen, da sie sonst nicht die beschriebene Auswirkungen auf die Übersetzerliste haben. Ferner sind diese Anweisungen wirkungslos, falls sie im Innern von Kommentaren auftauchen.

5. Datenaustausch mit in FORTRAN IV geschriebenen Haupt- und Unterprogrammen

Der PL/I Optimizing Compiler gestattet den Aufruf von FORTRAN-IV-Unterprogrammen (FUNCTION, SUBROUTINE) von PL/I-Programmen aus und bereitet PL/I-Prozeduren für den Aufruf durch FORTRAN-IV-Programme vor. Für den Einsatz dieser Einrichtung ist zu beachten, daß verschiedene PL/I-Datentypen und PL/I-Strukturen in FORTRAN IV keine Entsprechung besitzen. Hingegen können die unterschiedlichen Organisationen von mehrfachdimensionierten Feldern angepaßt werden.

- Ein FORTRAN-IV-Unterprogramm, das durch eine CALL-Anweisung oder eine Funktionsreferenz von einem PL/I-Programm aus aufgerufen werden soll, muß in diesem deklariert werden:

```
DCL identifier EXTERNAL ENTRY [(parameter-descriptor-list)]
    [VARIABLE] [RETURNS(attribute-list)]
    OPTIONS (FORTRAN [additional-options])
```

mit folgenden möglichen Optionen (additional-options):

```
INTER          NOMAP [(argument-list)]
RETCODE        NOMAPIN [(argument-list)]
               NOMAPOUT [(argument-list)]
```

und den argument-list-Eintragen:

```
ARGI[, ARGJ]... mit ganzen Dezimalzahlen i, j, ...;
```

dabei bewirkt INTER eine spezielle Behandlung der im FORTRAN-Programm auftretenden Unterbrechungen (interrupts), RETCODE die Verfügbarmachung eines Returncodes und NOMAP, NOMAPIN, NOMAPOUT die Unterdrückung der Manipulation der Felder bei der Parameterübergabe zwischen PL/I- und FORTRAN-IV-Programm.

Beispiel:

```

DCL FORTA ENTRY (BIN FIXED(31)...DEC FLOAT)
      OPTIONS (FORTRAN RETCODE INTER
              NOMAP(ARG2)
              NOMAPOUT(ARG3));
:
:
CALL FORTA (I,XM1,XM2,X);
RC = PLIRETV();

```

- Zur Vorbereitung eines PL/I-Programms für den Aufruf durch ein FORTRAN-IV-Programm müssen entsprechende Optionen in die PROCEDURE-Anweisung eingetragen werden:

```

entry-constant: PROC [(parameter-list)]
                  [RETURNS (attribute-list)]
                  OPTIONS (FORTRAN [additional-options]);

```

mit folgenden möglichen Optionen:

```

NOMAP [(argument-list)]
NOMAPIN [(argument-list)]
NOMAPOUT [(argument-list)],

```

mit den oben beschriebenen Bedeutungen.

Beispiel:

```

PL1A: PROC (X) RETURNS (DEC FLOAT)
      OPTIONS (FORTRAN NOMAPOUT(ARG1));

```

Zum Einsatz der Batch-Compiler

W. Bosse

Für die Programmiersprachen ALGOL W, Assembler, FORTRAN IV, PL/I und SNOBOL4 stehen am Rechenzentrum der Universität Münster Batch-Compiler zur Verfügung. Diese bieten im Vergleich zu den üblichen "großen" Sprachübersetzern mehrere Vorteile und sollten deshalb neben der obligatorischen Verwendung im Rahmen von Programmierkursen nach Möglichkeit auch als Testübersetzer bei jeder Programmentwicklung eingesetzt werden.

Die Batch-Compiler sind schnell in der Übersetzung und liefern bei aufgetretenen Programmfehlern eine ausführliche Beschreibung der Fehler. Sie werden tagsüber zu jeder vollen Stunde durch den Konsol-Operator aktiviert; alle bis dahin eingelesenen Batch-Läufe, die durch eine spezielle Angabe im Typ-Feld der Job-Karte gekennzeichnet werden müssen, können dann nacheinander abgearbeitet werden, ohne für jeden Job die sonst erforderlichen Dienste des Betriebssystems zu beanspruchen. Dadurch werden trotz des bestehenden Engpasses in der apparativen Ausstattung des Rechenzentrums recht günstige Bearbeitungszeiten für den Benutzer erzielt. Mehrere Programme, die in derselben Programmiersprache geschrieben worden sind,

lassen sich auch zu einem Job zusammenstellen. Außerdem wirken sich Batch-Läufe nicht negativ auf die Priorität der anschließend eingelesenen "normalen" Jobs aus.

Bei Batch-Läufen kann die Eingabe der Daten nur über den Kartenleser und die Ausgabe allein auf dem Drucker erfolgen; ferner bestehen einige Einschränkungen bezüglich der Rechenzeit, der zu druckenden Zeilen, des Arbeitsspeichers etc. Das ist aber kein Grund, bei größeren Anforderungen auf den Einsatz der Testübersetzer zu verzichten, denn jeder Batch-Compiler kann vom Benutzer auch über eine katalogisierte Prozedur in einem "normalen" Job aufgerufen werden. Dann bestehen prinzipiell keine Einschränkungen mehr, d.h. der Benutzer kann die verfügbaren Parameter seinen Wünschen entsprechend setzen und außerdem externe Speicher (Magnetplatte, -band) zur Ein-/Ausgabe von Daten verwenden.

Übersicht:

Programmier- sprache	Testübersetzer (Batch-Compiler)	Job-Typ (Batch-Lauf)	Katalogisierte Prozedur
ALGOL W	ALGOLW	ALGW	ALGOLW
Assembler	SPASM	SPAS	SPASM
FORTTRAN IV	WATFIV	WATF	WATFIV
PL/I	PL/C	PL/C	PLC
SNOBOL4	SPITBOL	SNOB	SPITBOL

Nachfolgend werden zu jedem Batch-Compiler alle für den Aufruf wesentlichen Einzelheiten angegeben. Dabei beschreiben die mit // beginnenden Zeilen Steuerkarten, die für den Aufruf des Übersetzers und die Programmausführung benötigt werden; diese Karten liegen im Locherraum des Rechenzentrums aus. Entsprechend ihren Eintragungen heißen sie Job-Karte, //ALGOLW-, //PL/C-, //SPASM-, //SPITBOL-, //WATFIV- und //DATA-Karte sowie Endkarte. In die Job-Karte muß der Benutzer einige persönliche Angabe eintragen:

jobnr die fünfstellige Rechennummer;

box die dreistellige Fachnummer (für die Druckausgabe);

name der Name des Programmierers;

xxx 0 bis 3 Zeichen (Buchstaben, Ziffern, #, \$, @) zur Benennung des Laufes.

Diese Angaben dürfen kein Leerzeichen (Blank) enthalten; nur die Eintragung JOB ist in Blanks eingeschlossen.

ALGOL-W

Version des Testübersetzers: 01JULY72

Sprachumfang: die Programmiersprache ALGOL W

Zusammenstellung eines Batch-Laufs:

```
//jobnrxxx JOB (jobnr,ALGW,box),name
//ALGOLW
```

(ALGOL-W-Programmkarten)

```
//DATA
```

(Datenkarten, falls vorhanden)

```
//ALGOLW
```

```
//DATA
```

```
//
```

in dieser Form können mehrere Programme (mit Daten) angefügt werden

ENDKARTE

Die //DATA-Karte wird nur dann benötigt, wenn Daten vorhanden sind.

Parameter auf der //ALGOLW-Karte:

T=sec maximale Ausführungszeit in sec (implizit 20, maximal 40);

P=pages maximale Anzahl der Seiten Druckausgabe (implizit 6, maximal 12)

Diese Parameter können auf der //ALGOLW-Karte ab Spalte 10 angegeben werden (ohne Leerzeichen); Komma ist Trennzeichen.

Steueranweisungen:

\$LIST oder \$NOLIST die nachfolgenden Programmkarten werden aufgelistet (implizit) bzw. nicht aufgelistet;

\$TITLE,"text" oder \$TITLE die Programmliste wird auf einer neuen Seite fortgesetzt, wobei text als Überschrift erscheint (maximal 30 Zeichen) bzw. keine Überschrift gegeben wird;

\$SYNTAX das Programm wird nur auf syntaktische Fehler geprüft, aber nicht ausgeführt;

\$STACK bei einem Fehler während der Syntax-Analyse des Programms wird der Inhalt des "Parsing Stack" ausgegeben;

\$NOCHECK spezielle Kontrollen beim Programmablauf entfallen (Index-Grenzen, nicht initialisierte Variablen, etc.);

\$DEBUG,m(n) die durch m ($0 \leq m \leq 4$) angegebenen Testhilfen werden für jedes Statement mit der Häufigkeit n ($n \geq 0$) verwendet.

Diese Steueranweisungen müssen jeweils auf einer Lochkarte in Spalte 1 beginnen und können in Verbindung mit den Programmkarten verwendet werden.

Testhilfen:

Mittels der Steueranweisung \$DEBUG können Testhilfen angefordert werden. Die Prozedur TRACE gestattet eine Veränderung der angegebenen Häufigkeit; Aufruf: TRACE(n). Außerdem können verschiedene Fehler-Ereignisse über implizit deklarierte Zeiger-Variablen gesteuert werden, die an die record-Klasse EXCEPTION gebunden sind (ENDFILE, OVFL, UNFL, DIVZERO, INTOVFL, INTDIVZERO, SORTERR, EXPERR, LNLOGERR, SINCOSEERR).

Dokumentation:

W. Bosse, "Einführung in das Programmieren mit ALGOL W",
Schriftensammlung des Rechenzentrums, Heft 14, 1975,
Anhang F (S.209-223)

J. Eve, "ALGOL W Reference Manual / ALGOL W Programmers
Guide", University of Newcastle upon Tyne, 1972

Beispiel eines Batch-Laufs:

```
//XA001NEU JOB (XA001,ALGW,P01),SOMMER
//ALGOLW
      (Programm)
//DATA
      (Daten)
//ALGOLW T=30,P=10
      (Programm)
//DATA
      (Daten)
//
```

ENDKARTE

Aufruf über eine katalogisierte Prozedur:

Prozedurname: ALGOLW

Speicherbedarf: mindestens 120 KBytes

Symbolische Parameter beim Aufruf:

MINUTES=min und/oder SECONDS=sec maximale Ausführungs-
zeit in min:sec (implizit 0:40);

PAGES=pages maximale Anzahl der Seiten Druckausgabe
für jedes Programm (implizit 12).

Beispiel eines Aufrufs:

```
//UUU01A JOB (UUU01,MINI,Q01),WINTER
// EXEC ALGOLW,MINUTES=1,SECONDS=0
//SYSIN DD DATA
      (Programme und Daten mit den //ALGOLW-
      und //DATA-Karten wie im Batch-Lauf)
```

/*

//

ENDKARTE

PL/C

Version des Testübersetzers: Release 7.5

Sprachumfang: eine Teilmenge der Programmiersprache PL/I; nicht verfügbar sind insbesondere Sprachelemente zur Listenverarbeitung, CONTROLLED- und BASED-Variablen, Preprocessor-Anweisungen und einige BUILTIN-Funktionen.

Zusammenstellung eines Batch-Laufs:

```
//jobnrxxx JOB (jobnr,PL/C,box),name
```

```
//PL/C
```

[PL/I-Programmkarten]

```
//DATA
```

[Datenkarten, falls vorhanden]

```
//PL/C
//DATA
//
```

In dieser Form können mehrere Programme (mit Daten) angefügt werden

ENDKARTE

Die //DATA-Karte wird nur dann benötigt, wenn Daten vorhanden sind.

Parameter auf der //PL/C-Karte:

TIME=(min,sec) [T] maximale Übersetzungs- und Ausführungszeit in min:sec (implizit 0:20, maximal 0:40);

PAGES=pages [P] maximale Gesamtzahl der zu druckenden Seiten (implizit 5, maximal 10);

LINES=lines [L] maximale Gesamtzahl der zu druckenden Zeilen (implizit 400, maximal 800);

LINECT=linecount [LC] Anzahl der Zeilen pro Seite bei der Programmliste (implizit 64);

ERRORS=(c,e) [E] das Programm soll nicht (weiter) ausgeführt werden, falls bei der Übersetzung mehr als c bzw. bei der Ausführung mehr als e Fehler auftreten (implizit c=20, e=10, maximal c=255, e=255);

FLAGE [FE] oder FLAGW [FW] es werden nur Error-Messages bzw. zusätzlich auch Warnings (implizit) gedruckt;

ATR [A] oder NOATR [NA] die Attribut-Liste wird ausgegeben bzw. nicht ausgegeben (implizit);

OPLIST [O] oder NOOPLIST [NO] die Liste aller gesetzten Optionen wird ausgegeben bzw. nicht ausgegeben (implizit);

XREF [X] oder NOXREF [NX] die Zuordnungstabelle wird erstellt bzw. nicht erstellt (implizit);

SOURCE [S] oder NOSOURCE [NS] das Programm wird aufgelistet (implizit) bzw. nicht aufgelistet;

Weitere Parameter werden in der Dokumentation beschrieben. Die gewünschten Parameter können auf der //PL/C-Karte ab Spalte 8 angegeben werden (ohne Leerzeichen); Komma ist Trennzeichen. Neben der ausführlichen Schreibweise kann auch die in Klammern angegebene Kurzform verwendet werden.

Steueranweisungen:

*OPTIONS ab Spalte 10 können für den folgenden Programmteil Parameter neu gesetzt werden (mit Ausnahme von ERRORS, LINES, PAGES und TIME);

*PROCESS trennt zwei externe Prozeduren; ab Spalte 10 können für die folgende Prozedur Parameter neu gesetzt werden (Ausnahmen wie bei *OPTIONS);

*INCLUDE m₁,m₂,...; die Member m₁,m₂,... werden in festgelegten Bibliotheken (z.B. SSP,PL1) gesucht, und der in ihnen enthaltene Text wird anstelle dieser Anweisung in das Programm eingefügt;

Diese Steueranweisungen müssen jeweils auf einer Lochkarte in Spalte 1 beginnen und können in Verbindung mit den Programmkarten verwendet werden.

Testhilfen:

Neben den in PL/I verfügbaren ON-Bedingungen und Bedingungspräfixen gibt es die Bedingung FLOW, die in Verbindung mit den Statements FLOW und NOFLOW ein (gesteuertes) Protokoll des Programmablaufs erzeugt. Analog dazu existieren für die Bedingung CHECK die Statements CHECK und NOCHECK. Ferner sind im PUT-Statement die Optionen OFF, ON, FLOW, SNAP, ALL, ARRAY und DEPTH möglich, und es können "Pseudokommentare" mittels des Parameters CMNTS zum Programmtest verwendet werden.

Dokumentation:

"User's Guide to PL/C", Cornell University, 1976
 (Software-Information Nr. 4 des Rechenzentrums, 1977)

Beispiel eines Batch-Laufs:

```
//XA001NEW JOB (XA001,PL/C,P01),SOMMER
//PL/C
    (Programm)
//DATA
    (Daten)
//PL/C TIME=40,PAGES=10,ATR,DPLIST,XREF
    (Programm)
//DATA
    (Daten)
//
ENDKARTE
```

Aufruf über eine katalogisierte Prozedur:

Prozedurname: PLC
 Speicherbedarf: mindestens 120 KBytes
 Symbolische Parameter beim Aufruf: keine

Beispiel eines Aufrufs:

```
//UUU01P JOB (UUU01,MINI,Q01),WINTER
// EXEC PLC
//SYSIN DD DATA
    (Programme und Daten mit den //PL/C-
    und //DATA-Karten wie im Batch-Lauf)
/*
//
ENDKARTE
```

SPASM

Version des Testübersetzers: Version 4

Sprachumfang: die Programmiersprache IBM System /360 Assembler (Level F); einige Einschränkungen beruhen insbesondere auf der Struktur von SPASM (Single Pass Assembler); System-Makros können nicht verwendet werden.

Zusammenstellung eines Batch-Laufs:

```
//jobnrxxx JOB (jobnr,SPAS,box),name
//SPASM
```

(Assembler-Programmkarten)

```
//DATA
```

(Datenkarten, falls vorhanden)

```
[ //SPASM
  |
  //DATA
  |
  // ]
```

In dieser Form können mehrere Programme (mit Daten) angefügt werden

ENDKARTE

Um die Ausführung eines Programms zu erreichen, muß die //DATA-Karte selbst dann angegeben werden, wenn keine Daten vorhanden sind.

Parameter auf der //SPASM-Karte:

LNPP=linecount Anzahl der Zeilen pro Seite (implizit 60);

ERR=code Fehlermeldungen werden nur dann ausgegeben, falls der Fehlercode mindestens gleich code ist (implizit 0);

COND=code das Programm wird nicht ausgeführt, falls Fehler mit einem Fehlercode auftreten, der mindestens gleich code ist (implizit 6);

GO oder NGO bei Programmfehlern wird die Ausführung erzwungen (implizit) bzw. nicht erzwungen;

ECOL oder NECOL "papiersparende" Druckausgabe wird gewünscht (implizit) bzw. nicht gewünscht;

PRNT oder NPRNT das Programm wird aufgelistet (implizit) bzw. nicht aufgelistet;

FIX oder NFIX Fixups werden aufgelistet (implizit) bzw. nicht aufgelistet;

XREF oder NXREF die Symbol-Zuordnungstabelle wird erstellt (implizit) bzw. nicht erstellt;

UMAP oder NUMAP die Using-Map wird ausgegeben (implizit) bzw. nicht ausgegeben;

ADMP oder NADMP nach der Assembly wird ein Programm-Dump ausgedruckt bzw. nicht ausgedruckt (implizit);

DUMP oder NDUMP nach der Programmausführung wird ein Dump ausgegeben (implizit) bzw. nicht ausgegeben;

BTRC oder NBTRC die Verzweigungstabelle wird ausgegeben (implizit) bzw. nicht ausgegeben.

Weitere Parameter werden in der Dokumentation beschrieben. Parameter können auf der //SPASM-Karte ab Spalte 9 ange-

geben werden (ohne Leerzeichen); Komma ist Trennzeichen.
Für jedes Programm ist die Rechenzeit auf 40 sec und die Druckausgabe auf 600 Zeilen beschränkt.

Steueranweisungen:

AOPTIONS $op_1, op_2, \dots$ als Operanden $op_1, op_2, \dots$ sind u.a. die Parameter FIX, XREF, UMAP oder deren Negation (Vorsilbe NO) möglich; weitere Parameter werden in der Dokumentation beschrieben;
DUMPOUT loc_1, loc_2 von dem durch loc_1 und loc_2 begrenzten Speicherbereich wird ein Dump ausgegeben;
INSTRACE loc_1, loc_2 nach Ausführung einer Instruktion, deren Adresse innerhalb von loc_1 und loc_2 liegt, wird das PSW ausgegeben;
INSTRACE =OFF beendet das Protokoll des Programmablaufs.
Diese Steueranweisungen werden in Verbindung mit den Programmkarten wie Assembler-Instruktionen angegeben.

Testhilfen:

Die vorgegebenen Makro-Instruktionen DUMPOUT, INSTRACE und MONITOR können zum Programmtest verwendet werden.

Dokumentation:

"SPASM Language Description and User's Guide", SCIP (SLAC), Stanford University, 1973

Beispiel eines Batch-Laufs:

```
//XA001ASS JOB (XA001,SPAS,P01),SOMMER
//SPASM
    {Programm}
//DATA
    {Daten}
//SPASM LNPP=68,ADMP
    {Programm}
//DATA
    {Daten}
//
```

ENDKARTE

Aufruf über eine katalogisierte Prozedur:

Prozedurname: SPASM
Speicherbedarf: mindestens 120 KBytes
Symbolische Parameter beim Aufruf:
SECONDS=sec maximale Ausführungszeit in sec für jedes Programm (implizit 40);
LINES=lines maximale Anzahl der Zeilen Druckausgabe für jedes Programm (implizit 600).

Beispiel eines Aufrufs:

```
//UUU01SAS JOB (UUU01,MINI,Q01),WINTER
// EXEC SPASM,LINES=300
//SYSIN DD DATA
    {Programme und Daten mit den //SPASM-
    und //DATA-Karten wie im Batch-Lauf}
/*
//
```

ENDKARTE

SPITBOL

Version des Testübersetzers: Version 2.0

Sprachumfang: die Programmiersprache SNOBOL4 (mit geringfügigen Einschränkungen); neben einigen neuen Funktionen gibt es auch bessere Eingabe/Ausgabe-Möglichkeiten.

Zusammenstellung eines Batch-Laufs:

```
//jobnrxxx JOB (jobnr,SNOB,box),name
```

```
//SPITBOL
```

```
[SNOBOL4-Programmkarten]
```

```
//DATA
```

```
[Datenkarten, falls vorhanden]
```

```
//SPITBOL
```

```
//DATA
```

```
//
```

in dieser Form können mehrere Programme (mit Daten) angefügt werden

ENDKARTE

Die //DATA-Karte wird nur dann benötigt, wenn Daten vorhanden sind.

Parameter auf der //SPITBOL-Karte:

LINECNT=linecount [L] Anzahl der Zeilen pro Seite bei der Programmliste (implizit 58).

Dieser Parameter kann auf der //SPITBOL-Karte ab Spalte 11 angegeben werden (ohne Leerzeichen).

Beim Aufruf über eine katalogisierte Prozedur sind zusätzlich noch folgende Parameter auf der //SPITBOL-Karte möglich (Komma ist Trennzeichen):

TIME=sec [T] maximale Übersetzungs- und Ausführungszeit in sec (implizit 40);

PAGES=pages [P] maximale Gesamtzahl der zu druckenden Seiten (implizit 10).

Neben der ausführlichen Schreibweise kann auch die in Klammern angegebene Kurzform verwendet werden.

Steueranweisungen:

-EJECT die Programmliste wird auf einer neuen Seite fortgesetzt;

-SPACE n oder -SPACE bewirkt n Leerzeilen bzw. eine Leerzeile in der Programmliste;

-TITLE text oder -STITLE text die Programmliste wird auf einer neuen Seite fortgesetzt, wobei text (innerhalb der Spalten 8-72 angegeben) als Titel bzw. als Untertitel erscheint;

-LIST oder -NOLIST die nachfolgenden Programmkarten werden aufgelistet (implizit) bzw. nicht aufgelistet;

-ERRORS oder -NOERRORS wenn beim Übersetzen des Programms Fehler entdeckt worden sind, wird die Ausführung erzwungen bzw. unterbunden (implizit).

WATFIV

Version des Testübersetzers: VIL4

Sprachumfang: die Programmiersprache FORTRAN IV (mit geringfügigen Einschränkungen); als Spracherweiterung steht insbesondere der Variablen-Typ CHARACTER zur Verfügung.

Zusammenstellung eines Batch-Laufs:

```
//jobnrxxx JOB (jobnr,WATF,box),name
//WATFIV
    (FORTRAN-IV-Programmkarten)
//DATA
    (Datenkarten, falls vorhanden)
```

```
//WATFIV
|
//DATA
|
//
```

In dieser Form können mehrere Programme (mit Daten) angefügt werden

ENDKARTE

Um die Ausführung eines Programms zu erreichen, muß die //DATA-Karte selbst dann angegeben werden, wenn keine Daten vorhanden sind.

Parameter auf der //WATFIV-Karte:

TIME=sec maximale Ausführungszeit in sec (implizit 20, maximal 40);

PAGES=pages maximale Anzahl der Seiten Druckausgabe (implizit 5, maximal 10);

LINES=linecount Anzahl der Zeilen pro Seite (implizit 64);

CHECK oder NOCHECK spezielle Kontrollen auf nicht initialisierte Variablen werden beim Programmablauf durchgeführt (implizit) bzw. nicht durchgeführt;

FREE entspricht CHECK und erzwingt auch bei Programmfehlern die Ausführung;

WARN oder NOWARN alle Warnings werden ausgegeben (implizit) bzw. nicht ausgegeben;

EXT oder NOEXT alle Extension-Mitteilungen werden ausgegeben (implizit) bzw. nicht ausgegeben;

LIST oder NOLIST das Programm wird aufgelistet (implizit) bzw. nicht aufgelistet;

LIBLIST oder NOLIBLIST der Programmtext von aus festgelegten Programmbibliotheken (z.B. SSP.WATFIV) automatisch eingefügten Unterprogrammen wird aufgelistet bzw. nicht aufgelistet (implizit).

Diese Parameter können auf der //WATFIV-Karte ab Spalte 10 angegeben werden (ohne Leerzeichen); Komma ist Trennzeichen.

Steueranweisungen:

//CHECK oder //NOCHECK

//WARN oder //NOWARN

//EXT oder //NOEXT die Parameter CHECK/NOCHECK,

WARN/NOWARN und EXT/NOEXT können innerhalb des Programms hiermit neu gesetzt werden;
 //EJECT die Programmliste wird auf einer neuen Seite fortgesetzt;
 //SPACE bewirkt eine Leerzeile in der Programmliste;
 //PRINTON oder //PRINTOFF die nachfolgenden Programm-
 karten werden aufgelistet (implizit) bzw. nicht aufgelistet;
 //ISNON aktiviert für die folgenden Statements die Ausgabe eines Protokolls des Programmablaufs (vor //ISNON muß mindestens ein ausführbares Statement stehen);
 //ISNOFF hebt die Wirkung von //ISNON auf.
 Diese Steueranweisungen müssen jeweils auf einer Lochkarte in Spalte 1 beginnen und können in Verbindung mit den Programmkarten verwendet werden.

Testhilfen:

Mittels der Steueranweisungen //ISNON und //ISNOFF kann ein (gesteuertes) Protokoll des Programmablaufs erzeugt werden. Ferner gibt es das Statement DUMPLIST, das wie ein NAMELIST-Statement aufgebaut ist und im Falle eines Programmabbruchs die Ausgabe der Werte aller benannten Variablen bewirkt, und das Statement ON ERROR GOTO, wodurch im Falle eines Programmfehlers zu einem speziellen Programmteil verzweigt werden kann, sowie die Unterprogramme TRAPS, OVERFL und DVCHK.

Dokumentation:

"WATFIV User's Guide", University of Waterloo, 1973

Beispiel eines Batch-Laufs:

```
//XA001FOR JOB (XA001,WATF,P01),SOMMER
//WATFIV
    {Programm}
//DATA
    {Daten}
//WATFIV TIME=30,LINES=72,NOEXT
    {Programm}
//DATA
    {Daten}
//
```

ENDKARTE

Aufruf über eine katalogisierte Prozedur:

Prozedurname: WATFIV
 Speicherbedarf: mindestens 120 KBytes
 Symbolische Parameter beim Aufruf: keine

Beispiel eine Aufrufs:

```
//UUU01WAT JOB (UUU01,MINI,Q01),WINTER
// EXEC WATFIV
//SYSIN DD DATA
    {Programme und Daten mit den //WATFIV-
    und //DATA-Karten wie im Batch-Lauf}

/*
//
```

ENDKARTE

PL/C Release 7.5

P. Lommel

Mit Beginn des Sommersemesters 1977 wird PL/C Release 7.5 die alte Version (7.1) des PL/I-Batch-Testübersetzers ersetzen. Folgende Erweiterungen sind nun verfügbar:

- Das PICTURE-Attribut und -Formatelement wurde implementiert.
- Implizite Konvertierung von Zahlen/Zeichenketten ist nun möglich. Der Übersetzer gibt jedoch einen Hinweis (Warning), wenn eine implizite Konvertierung auftritt.
- Einige Steuerkarten-Optionen wurden hinzugefügt.

Der "User's Guide to PL/C", der die Version 7.5 beschreibt, wird noch im Sommersemester 77 als Software-Information Nr. 4 vorliegen.

Drucken und Stanzen in ALGOL W

W. Bosse

Bei der verfügbaren Version des ALGOL-W-Compilers kann man zur Ausgabe von Werten neben den Standardprozeduren WRITE und WRITEON auch die Prozedur PUTF verwenden, bei der ein Ausgabe-Format in Form einer Text-Konstanten oder -Variablen anzugeben ist (solche Formate sind ähnlich aufgebaut wie in FORTRAN IV); außerdem gestattet PUTF auch die Stanzausgabe, allerdings nicht innerhalb eines Batch-Laufs.

Aufruf der Prozedur PUTF:

PUTF (DD-Name, Format, Wert-1, ..., Wert-n)

DD-Name

beschreibt das Ausgabe-Medium; anzugeben sind dabei

"SYSPRINT" für die Druckausgabe oder
"SYSPUNCH" für die Stanzausgabe.

Format

enthält als Text-Konstante oder -Variable die gewünschten Format-Codes.

Wert-1, ..., Wert-n

sind die nach diesem Format auszugebenden Werte.

Bei jedem Aufruf von PUTF wird eine neue Zeile bzw. Karte begonnen. Für die Druckausgabe sollte man nach Möglichkeit nicht PUTF und WRITE/WRITEON nebeneinander verwenden, um Fehler in der Reihenfolge zu vermeiden.

Eine Beschreibung der Format-Codes wird im Anhang F von

W. Bosse, "Einführung in das Programmieren mit ALGOL W",

Schriftenreihe des Rechenzentrums, Nr. 14, 1975

gegeben; allerdings sind die dort genannten DD-Namen nicht mehr gültig. Dieser Anhang ist noch in begrenzter Stückzahl bei Frau E. Geisler (Zi. 101) erhältlich.

FORTRAN-Programmeditierung

H. Pudlatz

Als Lesehilfe für unübersichtlich geschriebene FORTRAN-Programme stelle ich ein PL/I-Programm zur Verfügung, daß solche Programme optisch strukturiert, indem es die Statementnummern geordnet neuverteilt und DO-Schleifen einrückt, auf Wunsch auch die auf GOTO-Statements folgenden Programmteile bei Vorwärtssprüngen bis zu der angegebenen Statementnummer (vgl. Programm-Parameter).

Aus dem links angegebenen Programmanfang wird z.B. die rechts stehende Version:

17 READ(5,1,END=39) A,B		10 READ(5,20,END=40) A,B
1 FORMAT(20F4.0)		20 FORMAT(20F4.0)
DO 15 I=1,10		DO 30 I=1,10
15 A=A+B/I		30 A=A+B/I
GOTO 222		GOTO 50
39 C=A/10.		40 C=A/10.

Bis auf die älteren I/O-Befehle DEBUG AT, READ <S>,<Liste>, PRINT <S>,<Liste> und PUNCH <S>,<Liste> werden dabei alle FORTRAN-Statements bearbeitet, vorausgesetzt, daß das "bestimmende Schlüsselwort" (z.B. IF) und die zugehörigen Statementnummern auf derselben Karte oder Fortsetzungskarte abgelocht sind. Haupt- und Unterprogramme können in einem Lauf eingegeben werden; sie werden separat umgewandelt.

Programm-Parameter (Default-Werte in Klammern):

RENUM = YES NO	: neue Statementnummern gewünscht	(YES)
START =	Anfangsstatementnummer für Neunummerierung	(10)
INCR =	Inkrement für Statementnummern	(10)
SHIFT =	einzurückende Spaltenzahl bei DO-Schleifen	(3)
GO = YES NO	: 'SHIFT' auch bei Vorwärts-GOTO	(NO)
COPY = YES NO	: Eingabeprotokoll gewünscht	(YES)
LABEL = YES NO	: neuer Label in Spalte 77-80	(NO)
NAME = xxxx	: Name (max 4-stellig) in Spalte 73-76, Falls LABEL=YES	()
PUNCH = YES NO	: Stanzausgabe gewünscht	(NO)

Das Programm, das für die Umwandlung von 1000 FORTRAN-IV-Programmkarten ca. eine Minute benötigt, läuft in 80 KBytes und ist mit folgenden Steuerkarten aufzurufen, wobei die Parameter über den symbolischen Prozedurparameter PARME (hier nur beispielhaft) übergeben werden:

```
//      EXEC PLOE,OBJECT='SYSTEM.PROGRAM',MODULE=URZ04FST
//      PARME='START=1,INCR=1,GO=YES,COPY=NO'
//SYSDISK DD UNIT=WORK,SPACE=(TRK,(20,2))
//SYSIN  DD *
```

Das Programm gehört nicht zur vom Rechenzentrum gewarteten System-Software. Bei etwaigen Schwierigkeiten im Umgang mit dem Programm steht der Autor beratend zur Verfügung.

Lehrangebot des Rechenzentrums in SS 1977FORTRAN IV

Ziel dieser Veranstaltung ist es, die wichtigsten Sprachelemente der Programmiersprache FORTRAN IV zu vermitteln, so daß Aufgaben mittleren Schwierigkeitsgrades aus dem naturwissenschaftlichen Bereich selbständig gelöst werden können. Dabei sollen die speziellen Eigenschaften des am Rechenzentrum verfügbaren FORTRAN-Testübersetzers (WATFIV) besonders berücksichtigt werden.

Dozenten: Blanke, Reichel
Vorl.-Nr.: 220088, 220092
Hörsaal: M3, M4
Zeit: Mi 15-17, Beginn 27.4.77,
bzw. Mo 13-15, Beginn: 25.4.77

PL/I für numerische Anwendungen

Einführung in die Benutzung der universellen Programmiersprache PL/I für die Lösung von Problemen aus Mathematik, Statistik und Naturwissenschaften. Vorkenntnisse in Mathematik oder EDV sind nicht erforderlich.

Dozent: Mertz
Vorl.-Nr.: 220107
Hörsaal: M5
Zeit: Mi 15-17 plus 1 Std. Übungen n.V., Beginn: 27.4.77

PL/I für nichtnumerische Anwendungen

Es wird ähnlich wie in der Parallelveranstaltung eine allgemeine Einführung in die Programmiersprache PL/I gegeben. Beispiele werden aber vor allem aus dem nicht-mathematischen Bereich gewählt werden, um von daher kommende Verständnisschwierigkeiten zu vermeiden.

Dozent: Sturm
Vorl.-Nr.: 220111
Hörsaal: M6
Zeit: Mi 15-17, Beginn: 27.4.77

ALGOL W

ALGOL W ist eine einfache, aus wenigen Grundelementen bestehende Programmiersprache, die aufgrund ihres klaren Aufbaus besonders dem Anfänger das Erlernen der Methoden und Techniken der strukturierten Programmierung erleichtert. Sie eignet sich neben der Bearbeitung numerischer Probleme auch für Fragestellungen der Informatik.

Dozent: Ebert, J.
Vorl.-Nr.: 220126
Hörsaal: Seminarraum Rechenzentrum, Roxeler Str. 60
Zeit: Mo 15-17, Beginn: 25.4.77

Grundzüge des interaktiven Programmierens und die Dialogsprache APL/360

APL ist eine Programmiersprache für die Problemlösung im Dialog. Der Kurs wird in die Arbeit mit Dialogstationen einführen und mit der interaktiven Programmierung in APL vertraut machen.

Anmeldung bis zum 29.4.77 im Dispatch erforderlich.

Dozent: Meyer, H.

Vorl.-Nr.: 220145

Hörsaal: Seminarraum Rechenzentrum, Hittorfstr. 27

Zeit: Do 16-18 plus 2 Std. Übungen n.V., Beginn: 5.5.77

PL/I für Fortgeschrittene

An leichtformulierten Aufgabestellungen sollen Möglichkeiten der Programmiersprache PL/I demonstriert und diskutiert werden. Programmierkenntnisse in einer Programmiersprache werden vorausgesetzt. Es werden eigene Programmierübungen für die Dauer des Kurses erwartet.

Dozent: Zörkendörfer

Vorl.-Nr.: 220150

Hörsaal: Seminarraum Rechenzentrum, Roxeler Str. 60

Zeit: Mi 15-17, Beginn: 20.4.77

Kontrollsprachen des Betriebssystems

Jede Aufgabe, die mit der Rechenanlage durchgeführt werden soll, muß in Form eines Jobs vorliegen. Dazu sind neben der Bereitstellung von Programmen und Daten auch Kontrollinformationen für das Betriebssystem zu schreiben. In Einzelthemen sollen ausgewählte Beispiele zum Einsatz der Kontrollsprache des Betriebssystems behandelt werden:

Katalogisierte Prozeduren für die Sprachübersetzer, Parameter auf der JOB- und EXEC-Karte, Arbeiten mit Magnetplatten und -bändern, Utility-Prozeduren, Linkage Editor, etc.

Dozent: Bosse

Vorl.-Nr.: 220164

Hörsaal: M2

Zeit: Di 15-17, Beginn: 26.4.77

Computernumerik

Thema ist die Praxis des Rechnens mit einer Datenverarbeitungsanlage. Ausgehend von einer Darstellung der Rechnerarithmetik werden Probleme wie overflow/underflow, Fehleranalyse, Wahl der Genauigkeit, Wahl von Algorithmen und Erstellung zuverlässiger Programme behandelt. Dabei werden Kenntnisse in einer Programmiersprache vorausgesetzt.

Dozent: Ebert, U.

Vorl.-Nr.: 220179

Hörsaal: Seminarraum Rechenzentrum, Roxeler Str. 60

Zeit: Mi 10-12, Beginn: 20.4.77

Automatische Übersetzung: Formale Modelle und Algorithmen

In dieser Vorlesung sollen verschiedene formale Modelle zur Charakterisierung von Übersetzungsprozessen [Homomorphismen, Buchstaben-Übersetzungssysteme, Segment-Übersetzungssysteme, endliche Übersetzungsautomaten, syntax-gesteuerte Systeme und MARKOV-Systeme] vorgestellt und hinsichtlich ihrer Leistungsfähigkeit verglichen werden.

Dozent: Slaby
Vorl.-Nr.: 220198
Hörsaal: Seminarraum Rechenzentrum, Hittorfstr. 27
Zeit: Mo 14-16, Beginn: 25.4.77

Computer-Kartographie (Erstellen thematischer Karten auf dem Schnelldrucker und Plotter)

Ausgehend von der Anwendung von Histogrammen und Kartogrammen zur Darstellung quantitativer Ergebnisse in den Geowissenschaften werden Programme besprochen und eingesetzt, um thematische Karten auf dem Schnelldrucker und mit Hilfe des Plotters und des graphischen Displays zu erzeugen. Die Lehrveranstaltung wendet sich an Studierende der Geowissenschaften und wird als Praktikum durchgeführt. Voraussetzung ist die erfolgreiche Teilnahme an einem Programmierkurs des Rechenzentrums.

Dozenten: Pudlitz, Streit
Vorl.-Nr.: (191861)
Hörsaal: Seminarraum 310, Geographie
Zeit: Di 14-16, Beginn: 19.4.77

Optimierung

Es wird die Theorie der linearen und nichtlinearen Optimierung sowie praktische Verfahren zur Bestimmung von extremen Werten von Funktionen mehrerer Veränderlicher, wie etwa die Methode von Davidon/Fletcher/Powell behandelt. Neben der linearen Optimierung soll der konvexen Optimierung besonderer Raum gegeben werden.

Dozent: Werner
Vorl.-Nr.: (150540)
Hörsaal: M6
Zeit: Mo, Do 9-11, Beginn: 18.4.77

Seminar über Datenbanksoftware

Konzepte und Realisierungen von Datenbanksystemen verschiedener Provenienz (u.a. IMS, ADABAS, UDB) sollen vorgestellt und verglichen werden. Programmierkenntnisse sind erforderlich.

Dozenten: Werner, Mertz, Schmitt
Vorl.-Nr.: 220202
Hörsaal: Seminarraum Rechenzentrum, Roxelerstr. 60
Zeit: Mo 17-19, Beginn: 25.4.77

Seminar für Mathematische Software

Erarbeitung von Algorithmen für Optimierungsaufgaben ohne Restriktionen. Anwendungen auf Approximation nach der Methode der kleinsten Quadrate, auf optimale Steuerungen und auf Optimierungsaufgaben mit Restriktionen nach Straf- und Schranken-Funktions-Methoden.

Dozenten: Ebert, U., Hornung
Vorl.-Nr.: (150942)
Hörsaal: Seminarraum 1
Zeit: Di 9-11, Beginn: 19.4.77

Oberseminar über Angewandte Mathematik

In diesem Oberseminar berichten Mitarbeiter des Instituts und Gäste über neuere Ergebnisse der angewandten Mathematik und insbesondere der numerischen Mathematik.

Dozent: Werner
Vorl.-Nr.: (150957)
Hörsaal: Seminarraum 1
Zeit: Do 11-13, Beginn: 28.4.77

Leserforum

Hiermit möchte ich die Einführung regelmäßiger Output-Ausgabezeiten vorschlagen. Regelmäßige Output-Ausgabezeiten in stündlichem oder zweistündlichen Turnus gäben dem Benutzer die Möglichkeit, seine Zeit besser einzuteilen. Eine zusätzliche Anzeige über den Umfang der Warteschlange mit eventueller Angabe der voraussichtlichen Wartezeit für Jobs verschiedener Priorität in Form eines Aushangs bei der Jobannahme wäre ebenfalls eine große Hilfe.

T. Bettecken

Inst. f. Strahlenbiologie

Siehe Beitrag von K. Reichel auf Seite 3/4.

Kann das Rechenzentrum nicht jeden Samstag vormittag offen sein, unabhängig davon, wie es um den Rechner steht? Man könnte dann die stark frequentierten Locher benutzen und seinen Output abholen.

E. Nembach

Inst. f. Metallforschung

Das Rechenzentrum (Locherraum, Ausgabefächer) ist unabhängig davon, ob eine Job-Eingabe möglich ist, samstags vormittags von 8.00 bis 12.00 Uhr geöffnet. Nur wenn der vorhergehende Freitag ein Feiertag war, bleibt das Rechenzentrum geschlossen. An den anderen Werktagen wird das Rechenzentrum in Abänderung der in inforum Nr. 1 angegebenen Zeiten nun schon um 7.00 Uhr geöffnet. Dringende Locharbeiten sollten früh morgens

(7.00 - 9.00 Uhr) oder spät am Abend (nach 20.00 Uhr) erledigt werden, da zu diesen Zeiten erfahrungsgemäß der Andrang nicht so groß ist. K. Reichel

Für die Zusendung von inforum bedanke ich mich und möchte Ihnen mitteilen, daß auch ein Rechenzentrum an dieser Schrift interessiert sein kann, das einen anderen Maschinentyp betreibt. Wenn man über das Angebot von anderen Rechenzentren in dieser Weise informiert wird, lassen sich Diskussionen mit den eigenen Benutzern oder dem zuständigen ADV-Ausschuss oft günstiger führen.

H. Felsch
Rechenzentrum,
Universität Bielefeld

Ich beglückwünsche Sie zur Idee einer Informationsschrift für das Rechenzentrum.

H.J. Neuhaus
Englisches Seminar

Die Redaktion behält sich vor, Leserbriefe und Beiträge im Leserforum gekürzt wiederzugeben.

Das Leserforum ist Ihr Forum. Wir bitten Sie deshalb um rege Beteiligung.

Richten Sie Ihren Beitrag zum Leserforum bitte an die im IMPRESSUM angegebene Anschrift der Redaktion oder benutzen Sie das an der Job-Eingabe erhältliche Formblatt Angebungen und Fragen an das Rechenzentrum, das Sie ausgefüllt in den Briefkasten am Haupteingang des Rechenzentrums einwerfen können.

IMPRESSUM

Redaktion inforum

W. Bosse (Tel.: 490-2476)
J. Ebert (Tel.: 490-2607)
P. Lommel (Tel.: 490-2686)
R. Schmitt (Tel.: 490-2475)

Rechenzentrum der Universität
Roxeler Str. 60
4400 Münster

Auflage dieser Ausgabe: 500

Redaktionsschluß der nächsten Ausgabe: 15.6.1977

Programmgewerkschaft März 1977

J. Ebert, W. Schafmann

Die folgende Liste gibt eine Übersicht über die Verteilung der OS-Steps der im März 1977 gerechneten Benutzer-Jobs auf die einzelnen Programmiersprachen und (Hilfs-)Programme.

	BENUTZUNGEN	
1. FORTRAN-IV-G1-EXECUTE	5770	(17.61%)
2. FORTRAN-IV-G1-COMPILE	4522	(13.80%)
3. PL/I-OPT-EXECUTE	3537	(10.80%)
4. PL/I-OPT-COMPILE	3014	(9.20%)
5. LINKAGE EDITOR	2628	(8.02%)
6. SORT	1694	(5.17%)
7. SPSS	1208	(3.68%)
8. PL/I-F-EXECUTE	1164	(3.55%)
9. PLOT COPY	1012	(3.09%)
10. IEBGENER	809	(2.47%)
11. PL/I-F-COMPILE	758	(2.31%)
12. PL/I-BATCH	682	(2.08%)
13. RUMWTO	457	(1.39%)
14. FORTRAN-IV-BATCH	396	(1.20%)
15. ALGOL-W-COMPILE	381	(1.16%)
16. IEBCOPY	320	(0.97%)
17. ALGOL-W-BATCH	308	(0.94%)
18. UTILITY	273	(0.83%)
19. MPSX	248	(0.75%)
20. LOADER	233	(0.71%)
21. FORTRAN-IV-H-EXT-COMPILE	216	(0.65%)
22. ALGOL-W-EXECUTE	193	(0.58%)
23. TAPELIST	171	(0.52%)
24. RUMPGM	156	(0.47%)
25. RUMWTOR	148	(0.45%)
26. FORTRAN-IV-H-EXT-EXECUTE	142	(0.43%)
27. DELETE	110	(0.33%)
28. IEFBR14	109	(0.33%)
29. IEHLIST	91	(0.27%)
30. URZ13FMT	91	(0.27%)

Vergleicht man anhand dieser Liste die sprachbezogenen Job-Steps im Hinblick auf ihre Verteilung auf die Programmiersprachen, so ergibt sich für den Monat März 1977:

FORTRAN IV 52%, PL/I 43%, ALGOL W 4%.

Lieber Leser,

wenn Sie inforum regelmäßig beziehen wollen, bedienen Sie sich bitte des unten angefügten Abschnitts.

Hat sich Ihre Anschrift geändert oder sind Sie am weiteren Bezug von inforum nicht mehr interessiert, dann teilen Sie uns dies bitte auf dem vorbereiteten Abschnitt mit.

Bitte haben Sie Verständnis dafür, daß ein Versand außerhalb der Universität nur in begründeten Einzelfällen erfolgen kann.

Vielen Dank!

Redaktion inforum

An die
Redaktion inforum
Rechenzentrum
der Universität
Roxeler Str. 60
4400 Münster

Absender:

Name: _____

FB: _____ Institut: _____

Strasse: _____

Außerhalb der Universität: _____

☐ Ich bitte um Aufnahme in den Verteiler.

☐ Bitte streichen Sie mich aus dem Verteiler.

☐ Meine Anschrift hat sich geändert.

Alte Anschrift: _____

.....
(Datum)

.....
(Unterschrift)