KERNFORSCHUNGSZENTRUM

KARLSRUHE

September 1970

KFK 1253

Institut für Reaktorentwicklung

- M A P L I B -

Ein Programmsystem zur Bereitstellung von Stoffdaten für Rechenprogramme

U. Schumann

GESELLSCHAFT FÜR KERNFORSCHUNG M. B. H.

KARLSRUHE

KERNFORSCHUNGSZENTRUM KARLSRUHE

September 1970 KFK 1253

Institut für Reaktorentwicklung

- M A P L I B -

Ein Programmsystem zur Bereitstellung von Stoffdaten für Rechenprogramme

von

U. Schumann

Gesellschaft für Kernforschung mbH.,Karlsruhe

Zusammenfassung

MAPLIB (Material Properties Program Library) ist ein Programmsystem, das die Werte der Eigenschaften vieler Stoffe in der Form von Funktionen aufnehmen kann, um sie Rechenprogrammen in flexibler, effektiver, sicherer und transparenter Weise bereitzustellen. MAPLIB ist in FORTRAN IV geschrieben.
In MAPLIB sind eine Reihe von Konventionen für die Identifikation von Stoffen und Eigenschaften, die Form der Datenspeicherung und die physikalischen Einheiten der Daten festgelegt.
MAPLIB bietet ein flexibles Zugriffssystem an, das es gestattet, Anwendungsprogramme weitgehend unabhängig von den jeweiligen Stoffen und Eigenschaften zu programmieren.
In MAPLIB werden die vom Rechenprogramm gelieferten Parameterwerte, von denen die Funktionen abhängen, auf ihren Gültigkeitsbereich hin überprüft. Es sind zahlreiche Möglichkeiten vorhanden, auf Fehler zu reagieren.
MAPLIB verlangt, daß die aufzunehmenden Funktionen von Informationen über die in ihnen enthaltenen Daten begleitet werden. Es existiert ein Utility-Programm, das die Funktionen verwaltet, neue in MAPLIB aufnimmt und die Funktionen sowie Informationen über diese den Anwendern bereitstellt.

Abstract

MAPLIB (Material Properties Program Library) is a program system which is able to incorporate the values of the properties of many materials in the form of functions suitable for computer programs while providing flexibility, effectiveness, security and transparency. MAPLIB is written in FORTRAN IV. In MAPLIB some conventions are defined according to the identification of materials and properties, the form of the data storage and the physical units of the data. MAPLIB offers a flexible retrieval system, which allows to design user programs more or less independent of the special materials or properties.
In MAPLIB, the parameter-values delivered by the user program which are requested by the functions, are controlled with respect to their validity range. There are many possibilities available to react in the case of an error.
New functions, which shall be incorporated in MAPLIB, must be accompanied by information about the data described by the function.
A utility program exists which supervises the functions, incorporates new functions in MAPLIB and makes the functions and the information related to them available to the users.

Vorbemerkung

Bei dem Entwurf der hier beschriebenen Stoffdatenbank wurde auf Arbeiten von Herrn A. Pee aufgebaut. Die Anregung zum Entwurf des Programmsystems ging von Herrn E.G.Schlechtendahl aus, der auch ständig die Arbeit durch Beratung und anregende Diskussionen unterstützte.

Inhaltsverzeichnis

A b b i l d u n g e n

A n h a n g

1. Zweck und Aufbau von MAPLIB

1.1 Einführung

Für den Ingenieur spielen Stoffdaten eine entscheidende Rolle. Physikalisch kann man zwar viele Naturgesetze unabhängig von dem gerade betrachteten Stoff aufstellen, will man dagegen zu konkreten Aussagen kommen - wie z.B. in der technischen Auslegung eines Reaktors - muß man die speziellen Stoffe und deren Eigenschaften zahlenmäßig berücksichtigen.

In zunehmendem Maße werden diese Daten nicht nur bei Handrechnungen - am Schreibtisch mit Papier und Rechenschieber - benötigt, sondern innerhalb von Rechenprogrammen.
Hierfür wurden Daten bisher nur unsystematisch und in den verschiedenen Formen bereitgestellt (siehe jedoch [1]). Dieser Zustand verursachte eine Reihe von Mängel:

- Jeder, der Stoffdaten benötigt, muß sich weitgehend selbst die Daten zusammensuchen. Dabei entstehen viele isolierte "Privatbibliotheken". Es besteht die Gefahr der Doppelarbeit.

- Die mangelnde Normung der physikalischen Maßeinheiten der Stoffdaten, sowie der Parameter, ist eine häufige Fehlerquelle und macht einzelne Programme verschiedener Hersteller untereinander inkompatibel.

- Die Übergabe der Stoffdaten an das Rechenprogramm ist vielfach starr oder unhandlich.

- Die Anwendung von Stoffdaten führt zuweilen zu unsicheren Ergebnissen, wenn nicht überprüft wird, daß die Werte der Parameter im Gültigkeitsbereich der erstellten Funktionen liegen.

Es ist nun das Ziel von MAPLIB [2], alle diese Mängel auszugleichen. Man kann dieses Ziel etwas allgemeiner mit den von Hoare [3] geprägten Schlagworten

S I C H E R H E I T

E F F E K T I V I T Ä T

F L E X I B I L I T Ä T

kennzeichnen.
Hier wäre noch das Schlagwort

T R A N S P A R E N Z

hinzuzufügen, wie später erläutert wird (Kap.1.6).

1.2 Die wesentlichen Elemente von MAPLIB

MAPLIB besteht im wesentlichen aus vier Elementen (vergl.Abb.1).

1. Den integrierten Daten und den zugehörigen Informationen.
2. Konventionen für die Form der Daten
3. Einem Zugriffssystem zum flexiblen Abruf der Daten und zur Fehlerkontrolle.
4. Einem Utility-Programm
   - zur Erweiterung des Datenbestandes und zur Erstellung von Teilsystemen
   - zur Verwaltung des Datenbestandes
   - zur Dokumentation der Daten

Im Rahmen dieses Berichtes liegt der Schwerpunkt auf der Beschreibung der drei letzten Elemente. In Abb.1 sind die Dateien und Operatoren, auf die sich dieser Bericht bezieht, doppelt umrandet. Es ist dem Kreis der Benutzer vorerst selbst überlassen, die ihn interessierenden, aber eventuell fehlenden Stoffeigenschaften in das System einzubauen, so daß diese dann auch späteren Benutzern zur Verfügung stehen.

1.3 MAPLIB - Konventionen

1.3.1 Darstellung der Daten in der Form von Fortran-Funktionen

Üblicherweise werden Materialeigenschaften in Tabellen dargestellt. Diese Darstellungsweise ist für Computerrechnungen

schlecht geeignet, da Interpolationen erforderlich werden. Dagegen ist die Darstellung der Daten in einem Algorithmus sowohl bezüglich der Zugriffszeit, als auch bezüglich des Speicherbedarfs oftmals von Vorteil. Aus diesem Grunde wurde für die Darstellung der Material-Eigenschaftsdaten die Form der Funktion gewählt. Diese Funktion liefert jeweils für eine Reihe von Parameterwerten eine Date.
Die Funktionen werden als FORTRAN-FUNCTION programmiert, da FORTRAN die am meisten verbreitete Programmiersprache für technische Berechnungen ist.

1.3.2 Namensgebung

Die einzelnen Datenfunktionen sind charakterisiert durch den beschriebenen Stoff und die Eigenschaft. Die Datenfunktionen werden daher identifiziert durch die Verkettung zweier Symbole, von denen das eine die Eigenschaft und das andere den Stoff charakterisiert.

- Stoffe werden mit bis zu 4 alphamerischen Zeichen gekennzeichnet
- Eigenschaften werden mit 2 alphamerischen Zeichen gekennzeichnet, von denen das erste alphabetisch sein muß.

Zur einfachen Identifikation der Symbole wurden eine Reihe von Regeln festgelegt (siehe Anhang 1).
Die jeweilige Funktion erhält dann als Namen die Kombination des Eigenschafts- mit dem Stoffsymbol.
Z.B. ist

ES	das Symbol für die Eigenschaft Entropie
NAV	das Symbol für den Stoff Natrium in der Dampfphase
ESNAV	ist der Name der Funktion Entropie von Natrium-Dampf

Wenn dieses Symbol für einen Stoff bzw. eine Eigenschaft einmal definiert ist, müssen in der Regel die Namen aller neu in MAPLIB zu integrierenden Datenfunktionen mit diesen Symbolen

gebildet werden. Da diese starre Festlegung zuweilen nachteilig sein kann, gibt es jedoch auch Ausweichmöglichkeiten; siehe hierzu Kap.1.7.

1.3.3 Maßeinheiten

In MAPLIB wurde festgelegt, daß alle Maßeinheiten im M.K.S.A.K. (Meter Kilogramm Sekunde Ampére Grad-Kelwin) - System zu definieren sind [4]

z.B. Enthalpie in Nm/kg oder J/kg
Druck in N/m^2
Temperatur in ^oK

Sicher haben diese Einheiten bei Handrechnungen einige Nachteile. So ist es wohl unhandlich mit 10^5 N/m^2 statt ca. 1 ata zu rechnen. In Rechenprogrammen spielt dies keine Rolle. Der Vorteil der Einheitlichkeit dürfte - zumal nach einer gewissen Gewöhnung - den Nachteil etwas weniger handlicher Größenordnungen weit überwiegen. Dies zudem deshalb, weil bei Verwendung eines einheitlichen Einheitensystems alle Gleichungen wie Größengleichungen geschrieben werden können. Für Umrechnungsfaktoren siehe u.a. [5].

1.4 Zugriffssystem

Es ist eine Erfahrung bei der Erstellung von Problemprogrammen für nukleare oder allgemeine technische Untersuchungen, daß in zahlreichen Anwendungen bereits bei der Programm-Erstellung fixiert ist, mit welchen Stoffen und Eigenschaften das Programm zu rechnen hat.
In vielen anderen Fällen aber läßt sich ein und dasselbe Programm von seinem prinzipiellen Aufbau her für beliebige Stoffe oder Eigenschaften programmieren. Es wäre also wünschenswert, nicht schon zur Programmierzeit Stoff und Eigenschaft starr festlegen zu müssen, sondern diese, z.B. durch die Eingabe, zur Ausführungszeit zu bestimmen.
Diese Möglichkeit bietet MAPLIB in flexibler Weise. Der Zugriff zu den Datenfunktionen kann in drei verschiedenen Ebenen erfolgen.

Je höher die Ebene liegt, um so flexibler ist der Zugriff. (Vergl. Abb.3).
Auf der untersten Ebene ist die Funktion direkt mit ihrem Namen aufzurufen. Stoff und Eigenschaft sind hier also starr fixiert. Als Argumente müssen lediglich die Funktionsparameter übergeben werden. Auf den höheren Ebenen ruft man die Funktion nicht direkt, sondern über eine sogenannte Masterfunktion auf. Hier sind Stoff- und Eigenschaftssymbole zusätzlich zu den Funktionsparametern als variable Argumente zu übergeben.

Die Namensgebung dieser Masterfunktionen ist so normiert, daß an der Stelle des variablen Eigenschafts- oder Stoffsymbols 2 bzw.4 Füllzeichen stehen. Im Regelfall wird als Füllzeichen das Dollar-Zeichen gewählt.
Auf der mittleren Ebene werden Eigenschafts- und Stoff-Masterfunktionen unterschieden, je nachdem ob das Eigenschafts- oder Stoffsymbol im Masternamen fixiert ist. Das zweite Symbol muß in Form einer Zeichenkette im ersten Argument angegeben sein.

Die Masterfunktion der obersten Ebene erwartet sowohl das Eigenschaftssymbol als auch das Stoffsymbol als Argument. Mit dieser Funktion kann man zu jeder integrierten Datenfunktion zugreifen. Für weitere Erläuterungen und Beispiele sei auf die Benutzungsanleitung verwiesen.
Mit diesem Zugriffssystem soll die Forderung nach "Flexibilität" erfüllt werden.

1.5 Fehlerkontrollen

In diesem Abschnitt wird beschrieben, wie die Forderung"Sicherheit" erfüllt wird. Beim Aufruf einer Datenfunktion muß vom Benutzer mindestens ein Parameterwert an die Funktion übergeben werden. Für diese Parameterwerte soll der Funktionswert berechnet werden und an das aufrufende Programm zurückgegeben werden.
Ein fehlerhafter Aufruf einer Datenfunktion kann zahlreiche Folgen haben: vom Jobabbruch bis zum falschen Ergebnis. Zur

Feststellung der Fehlerquellen und zur Vermeidung der nachteiligen Folgen werden in der Regel von MAPLIB bei jedem Funktionsaufruf im wesentlichen die folgenden drei Kriterien überprüft:

a) ist die gewünschte Funktion in MAPLIB integriert?

b) wird die erwartete Anzahl der Parameter übergeben?

c) liegen die Werte der Parameter in dem Gültigkeitsbereich der Funktion?

Sicherheit geht jedoch im allgemeinen auf Kosten von Effektivität: Fehlerkontrollen kosten Rechenzeit. Bei einfachen Datenfunktionen dauert die Überprüfung der Argumente länger als die Berechnung des Funktionswertes. Dies kann zu einem schwerwiegenden Nachteil werden, wenn die Funktionen im Rahmen von Iterationen sehr häufig aufgerufen werden müssen.
In die Datenfunktionen werden daher Statements eingebaut, die es ermöglichen, von außen, durch Setzen eines bestimmten Steuerparameters, die Anweisungen, die die Fehlerkontrollen durchführen, zu überspringen. Der Benutzer kann sich also von Fall zu Fall in seinem Programm für "Sicherheit" oder "Effektivität" entscheiden.

Wenn MAPLIB einen Fehler feststellt, druckt es im Standardfall eine ausführliche Fehlermeldung in englischem Klartext aus und bricht dann die Ausführung des Programms ab. Über eine Reihe von Steuer-Parametern kann der Benutzer jedoch diese Standardreaktion des Systems in vielfacher Richtung abändern.
Das System bietet dann über mehrere Indikator-Parameter dem Benutzer zusätzlich die Möglichkeit festzustellen, ob und was für ein Fehler aufgetreten ist, so daß das Programm dann jeweils entscheiden kann, wie es sich weiterverhalten soll. Abb.4 zeigt die für Fehlerkontrollen und Zugriff maßgeblichen Programmbestandteile von MAPLIB, sowie den Informationsfluß. Für Einzelheiten sei auf die Benutzungsanleitung verwiesen.

1.6 Dokumentation

Den Schlagworten von Hoare ist noch ein viertes hinzuzufügen:

T R A N S P A R E N Z

Der Benutzer soll sich leicht darüber informieren können, mit welchen Stoffdaten er rechnet. Dies ist erforderlich, weil das System zu konzipiert ist, daß jeder aus dem Benutzerkreis von MAPLIB Funktionen, die in MAPLIB fehlen, selbst erstellen und in MAPLIB einbringen kann. Die Qualität der neuen Funktionen kann für die Probleme des Erzeugers ausreichend gewesen sein; für andere Aufgaben dagegen nicht. Jeder Benutzer trägt die Verantwortung für die Verwendung ausreichend genauer und moderner Daten. Um den Benutzer die Beurteilung zu ermöglichen, müssen daher die Funktionen von Informationen begleitet sein. Diese Information ist den Funktionen bei ihrer Integration in MAPLIB über besonders normierte Kommentarkarten mitzugeben. Die verschiedenen Informationsarten besitzen eine Identifikation, die in den ersten Spalten der Kommentarkarten abzulochen ist. Nach dieser Identifikation erfolgt die eigentliche Information im Anschluß auf den Kommentarkarten. Vergleiche hierzu Abb.3. Eine Tabelle der genannten Identifikationen befindet sich in Anhang 2.
Aufgrund der so an MAPLIB übermittelten Informationen kann das System über die integrierten Funktionen Auskunft geben. Dies wird von dem im folgenden Kapitel beschriebenen Utility-Programm ausgeführt.

1.7 Das MAPLIB-Utility-Programm

Das MAPLIB-Utility-Programm hat vor allem drei Aufgaben:

1. Einbau neuer Funktionen in MAPLIB
2. Ausgabe von Informationen
3. Sonderaufgaben

Zu 1)

Hier sind folgende Arbeiten durchzuführen:

a) Vervollständigung der Funktionen mit den Fehlerkontroll - routinen

b) Überprüfung der Information und eventuelle Vervollständigung mit Standardinformationen

c) Erweiterung der Masterfunktion

d) Aufbewahrung der Fortran-Karten

e) Erzeugung eines Load-Moduls für diese Funktion und Ablage des Load-Moduls auf einem speziellen partitional Data Set (mit Hilfe des Compilers und Linkage Editors)

Zu 2)

Auf Wunsch kann sich jeder Benutzer ausgeben lassen:

a) Eine Liste der Stoffsymbole mit Erläuterungen

b) Eine Liste der Eigenschaftssymbole mit Maßeinheitenangabe und Erläuterungen

c) Eine Liste der Funktions-Parametersymbole mit Maßeinheiten-angabe und Erläuterungen

d) Eine Tabelle der integrierten Funktionen mit den zugehörigen Parametern sowie eine Liste der System-Routinen

e) Listen der Masterfunktionen (einige oder alle)

f) Listen der integrierten Funktionen mit allen enthaltenen Kommentarkarten (einige oder alle)

Zu 3)

In der bisher beschriebenen Form hat MAPLIB noch drei wesentliche Nachteile:

a) Der Aufruf der Masterfunktion der obersten Ebene - "$$$$$$" - bewirkt, daß alle im MAPLIB integrierten Funktionen in das auszuführende Programm vom Linkage Editor eingebaut werden und somit Kernspeicherplatz belegten, wenn nicht spezielle Oberlay-Anweisungen gegeben wurden. Pro Funktion werden ca.

1000 Kernspeicher-Bytes benötigt.

Nun werden unter den Funktionen viele sein, für die sich der Benutzer nicht interessiert. Ein Teil der Funktionen ist für ihn also unnötiger Ballast.

b) Der zweite Nachteil besteht in der relativ starren Namensgebung der Funktionen. Die hier festgelegten Namen könnten mit Variablen- oder Funktionsnamen kollidieren, die der Benutzer bereits vorgesehen hat.

c) Ein dritter Nachteil besteht in der Abhängigkeit von Datasets, die nicht vom Benutzer selbst verwaltet werden und die durch Programm- oder Systemfehler zerstört werden können.

Zum Ausgleich dieser Nachteile besitzt der Benutzer mit dem Utility-Programm die Möglichkeit, sich eine maßgeschneiderte Privat-Stoffdatenbibliothek zuzulegen:

- er kann aus dem Gesamtbestand die ihn nicht interessierenden Funktionen löschen
- er kann die Namensgebung der Masterfunktionen so verändern, daß anstelle des Standard-Füllzeichens "$" ein beliebiger anderer Buchstabe eingesetzt wird
- er kann Datenfunktionen mit einem beliebigen anderen Namen versehen, der bei der Eingabe in das MAPLIB-Utility-Programm als Synonym zu dem aus den Konventionen folgenden Namen zu deklarieren ist
- er kann sich die Quell-Programme ausgeben lassen und als Fortran-Kartenpaket oder als übersetzte Programme auf Bändern, Karten oder anderen Datenträgern aufbewahren

Dies sind weitere Beiträge zur Forderung nach "Flexibilität".

1.8. Schluß

Mit MAPLIB steht somit ein Programmsystem bereit, das in der Lage ist, große Datenmengen aufzunehmen und diese Programmierern von technischen Anwendungsprogrammen in transparenter, sicherer, flexibler und effektiver Form bereitzustellen.

2. Benutzungsanleitung

2.1 Anwendung der MAPLIB-Bibliotheks-Routinen

(Angaben über Data-Sets und Job-Control Language sind auf die IBM 360/65 im KFZ-Karlsruhe abgestimmt)

In diesem Kapitel werden die Vorschriften für den Aufruf der Datenfunktionen und die Anwendung der Fehlersteuerroutinen beschrieben.
Über eventuelle Änderungen dieser Beschreibung kann sich der Benutzer im Rahmen eines FORTRAN-Programms durch Aufruf der Routine $NEWS mit dem Statement

```
CALL   $NEWS (NØUT)
```

informieren, wobei mit NOUT die Druckeinheit (Standard:NOUT=6) zu spezifizieren ist.

2.1.1 Aufruf der Datenfunktionen im Normalfall

Im weiteren bedeuten:

STOF	:	Stoffsymbol
EG	:	Eigenschaftssymbol
p_i	:	Parametername oder -konstante (REAL)
n	:	Anzahl dieser Parameter
s	:	Variable oder Kostante, deren Inhalt ein Stoffsymbol ist (INTEGER oder REAL mit der Länge 4 Bytes). Besteht das Stoffsymbol aus weniger als vier alphamerischen Zeichen, so ist es rechts mit Blanks (Leerstellen) auf vier Zeichen zu ergänzen. Das Stoffsymbol muß also linksbündig in s stehen.
e	:	Variable oder Kostante, deren Inhalt ein Eigenschaftssymbol ist (INTEGER oder REAL mit mindestens der Länge 2 Bytes).
x	:	Wert einer Materialeigenschaft (REAL)

Der Normalfall der Benutzung von MAPLIB besteht darin, daß der Benutzer den Wert einer Materialeigenschaft für bestimmte Werte der Parameter p_i (i=1,...,n) einer Variablen x in

seinem Programm zuweisen möchte, indem er die entsprechende Datenfunktion aufruft.
Dies setzt voraus:

- Für den verlangten Stoff und für die verlangte Eigenschaft ist in MAPLIB eine Funktion integriert.
- Der Benutzer liefert die richtige Zahl n der Parameter in der erwarteten Reihenfolge.(Die richtige Reihenfolge kann von MAPLIB nicht überprüft werden!)
- Die Werte der Parameter liegen im Gültigkeitsbereich der Funktion.
- Function-, Subroutinen-oder COMMON-Namen des Benutzer-Programms stimmen nicht mit entsprechenden Namen der MAPLIB-Routinen überein. (Zur Vermeidung unbeabsichtigter Übereinstimmungen mit den MAPLIB-System-Routinen sollte daher der Benutzer keine Namen verwenden, die mit einem Dollar-Zeichen beginnen oder enden. Bei Überschneidungen mit den Namen der Datenfunktionen besteht die Möglichkeit, die Standardnamen durch Synonyme zu ersetzen).
- Der Linkage-Editor oder Loader ist in der Lage, die gewünschten Routinen in den Load Module des Benutzer-Programms einzubauen. Siehe hierzu Kap.2.1.4

Der Benutzer kann sich anhand der Informationsausgaben des MAPLIB-UTILITY-Programms davon überzeugen, ob die ersten vier Voraussetzungen erfüllt sind; für die letzte hat er selbst zu sorgen.

Wenn diese Voraussetzungen gegeben sind, kann die gewünschte Datenfunktion auf drei verschiedenen Zugriffsebenen aufgerufen werden und das Ergebnis der Rechnung der Variablen x zugewiesen werden:

1) Oberste Ebene:

$$x = \$\$\$\$\$\$ (e, s, p_1, p_2, \ldots, p_n)$$

2) Mittlere Ebene:

$$x = \$\$STOF (e, p_1, p_2, \ldots, p_n)$$

Eigenschaftsebene

$$x = EG\$\$\$\$ (s, p_1, p_2, \ldots, p_n)$$

3) Unterste Ebene

$$x = EGSTOF (p_1, p_2, \ldots, p_n)$$

Diese Anweisungen gelten für die Standardbibliothek; vergl. jedoch Kap.2.1.3 .
Am Schluß des Benutzerprogramms sollte unmittelbar vor der Stop-Anweisung die Subroutine $STOP aufgerufen werden:

```
.....
.....
CALL $ STOP
STOP
END
```

Dieser Aufruf bewirkt die Ausgabe von MAPLIB-Nachrichten (z.B.über Änderungen) sowie einer Fehlerstatistik.

Beispiel:

```
      INTEGER   STØFF/'HEVb'/
      INTEGER*2   EIG/'ES'/
        T = 400.
        P = 1.E6
C     AUFRUF AUF DER UNTERSTEN EBENE
      X = ESHEV(T,P)

C     AUFRUF AUF DER MITTLEREN EBENE
C     STOFF-EBENE
      x = $$HEV(EIG,T,P)

C     EIGENSCHAFTS-EBENE
      x = ES$$$$(STØFF,T,P)
```

```
C   AUFRUF DER OBERSTEN EBENE
    x = $$$$$$(EIG,STØFF,T,P)
        .
        .
        .
        .
    CALL $ STOP
    STOP
    END
```

In diesem Beispiel hat ein Benutzer sich nach allen möglichen Methoden den Wert der Entropie von Helium für die Temperatur 400°K und den Druck 1•E6 N/M**2 von MAPLIB berechnen lassen und seiner Variablen x zugewiesen. Statt über die INTEGER-Statements kann er die Symbole HEVb und ES z.B. durch Einlesen den Variablen STOFF und EIG zuweisen. Er kann die Symbole auch direkt als Zeichenketten-Argument übergeben:

```
x = $$$$$$('ES','HEVb',400.,1•E6)
```

2.1.2 Reaktionen des Systems und mögliche Aktionen des Benutzers für den Fall eines Fehlers

Falls eine der im vorigen Kapitel genannten Voraussetzungen beim Aufruf der Funktion nicht erfüllt war, erhält der Benutzer vom MAPLIB-System eine Fehlermeldung.
Die Fehlermeldung erscheint im normalen Benutzungsfall in englischem Klartext auf Einheit 6 (normale Drucker-Einheit an der IBM 360/65); nach Feststellung des ersten Fehlers wird die Ausführung des Programms abgebrochen.
Es sind jedoch Möglichkeiten vorhanden, mit denen die Behandlung dieser Fehlernachrichten modifiziert und der Jobabbruch vermieden werden kann.
Hierzu sind die verschiedenen Fehlermöglichkeiten nach ihrer Ursache und Konsequenz zu differenzieren. Es werden 5 verschiedene Fehlerbereiche unterschieden, wie Tab.1 zeigt.

Tab.1: Fehlerbereiche

Fehler-bereich	Erläuterung	Ergebnis
1	Die Zahl der vom Benutzer angelieferten Parameter ist größer als die Bibliotheks-Funktion verlangt. Die zuviel angelieferten Parameter werden nicht berücksichtigt.	Funktions-wert (evt.falsch)
2	Der Wert eines Parameters liegt außerhalb des Gültigkeitsbereiches der Funktion; die Funktion ist jedoch für diesen Wert mathematisch noch definiert, MAPLIB liefert ein Rechenergebnis, das physikalisch falsch sein kann.	Funktions-wert (unsicher)
3	Für den angelieferten Parameterwert ist die Funktion mathematisch oder physikalisch nicht definiert. MAPLIB gibt den Standard-wert 1 zurück.	1.
4	Die Bibliotheksfunktion erwartet mehr Para-meter, als der Benutzer anliefert. Der Funktionswert kann nicht berechnet werden. MAPLIB liefert den Standardwert 1.	1.
5	Für die spezifizierte Eigenschaft und für den Stoff ist in MAPLIB keine Funktion integriert. MAPLIB liefert den Standard-wert 1.zurück.	1.

(Fehler 5 kann vom MAPLIB-System nur bei Aufruf der Funktion über die höheren Ebenen entdeckt und abgefangen werden; bei direktem Aufruf der Funktion wird der Fehler von Loader oder Linkage-Editor bemerkt und führt zum Abbruch der Programm-ausführung.)

Die in der Tab.2 angegebenen Steuerparameter definieren die Systemreaktionen bei Entdeckung eines Fehlers. Sie können mit den nachfolgenden Routinen vom Benutzer beliebig modifiziert werden:

Tab.2: Steuerparameter

Parameter -Name	Erläuterung	Standard-wert
NFILE	FORTRAN-Filenummer der Fehlernachrichtendatei	6 (SYSØUT=A)
NFALBE	In den Fehlerbereichen 1-4 zugelassene Gesamt-Fehlerzahl. Erst beim (NFALBE+1)-ten Fehler in einem beliebigen Fehlerbereich 1 ÷ 4 wird die Programmausführung abgebrochen.	0
NIRRE	Dito, nur für Fehlerbereich 5; siehe KSTP5	0
NØPR1 NØPR2 NØPR3 NØPR4 NØPR5	NØPRi ist die Anzahl der Fehler, die im Fehlerbereich i ohne Nachrichten zugelassen sind. Erst ab NØPRi+1 werden den Nachrichten geschrieben	alle:0
KPR1 KPR2 KPR3 KPR4 KPR5	KPRi ist die Anzahl der Fehlernachrichten des Fehlerbereiches i, die gedruckt werden sollen. Vom (KPRi+1)-ten Fehler an werden keine Nachrichten mehr geschrieben	alle:10
KSTP1 KSTP2 KSTP3 KSTP4 KSTP5	KSTPi ist die Anzahl der zugelassenen Fehler im Fehlerbereich i. Beim (KSTPi+1)-ten Fehler wird das Programm abgebrochen. KSTP5 = NIRRE	alle:0
TEST	LOGICAL, wenn TEST=•FALSE.gesetzt wurde, werden die Fehlerkontrollen - soweit in den Funktionen vorgesehen - nicht ausgeführt	.TRUE.

Es existieren folgende Routinen, um einige oder alle Parameter in beliebiger Weise zu modifizieren; angegeben ist die Anweisung, mit der die Routinen aufgerufen werden. Die Routinen übernehmen hierbei die Werte der bezeichneten Parameter:

```
CALL  $FILE(NFILE)
CALL  $INDEX(NFALBE,NIRRE)
CALL  $CNT(i,NØPRi,KPRi,KSTPi)
CALL  $NØPRT(i,NØPRi)
CALL  $END(i,KPRi)
CALL  $NSTØP(i,KSTPi)
CALL  $TEST(TEST)
CALL  $ERROR(NFILE,NFALBE,NØPR1,NØPR2,NØPR3,NØPR4,NØPR5,KPR1.
      KPR2;KPR3,KPR4,KPR5,KSTP1,KSTP2,KSTP3,KSTP4,KSTP5,TEST)
```

Durch das Statement

```
CALL  $FILE(4)
```

kann man erreichen, daß die Fehlermeldungen nicht mitten in der übrigen Ausgabe erscheinen, sondern im Anschluß ausgedruckt werden. Allerdings muß dazu eine DD-Karte im GO-Step vorhanden sein, z.B.

```
//G.FT04F001bDDbSYSØUT=A,DCB=*.FT06F001,
//   b   SPACE=(TRK,(3,1))
```

Neben diesen Steuerparametern, die die Reaktionen des Systems auf Fehler definieren, gibt es Indikator-Parameter (siehe Tab.3) die dem Benutzer Auskunft darüber geben, ob und wieviele Fehler in den verschiedenen Bereichen aufgetreten sind.

Tab. 3: Indikatorparameter

Parameter -Name	Erläuterungen	Anfangs- wert
LSTAT	Statusindikator (LOGICAL*4). Der Statusindikator wird FALSE gesetzt, wenn der erste Fehler auftritt.	.TRUE.
IFBER	Nummer des höchsten Fehlerbereiches	0
IF1 IF2 IF3 IF4 IF5	IFi gibt die Anzahl der entdeckten Fehler im Bereich i an	alle:0
LTEST	Der Indikator LTEST (LOGICAL*4) gibt an, ob der Steuerparameter TEST .TRUE. oder .FALSE. ist	TEST

Diese Parameter erhalten den Anfangswert bei Beginn der Programm-Ausführung und nach jeder Abfrage.
Die Abfrage ist nach folgenden Aufrufen möglich:

```
LSTAT   =   STAT$(DUMMY)
LTEST   =   TEST$(DUMMY)
```

(LSTAT und LTEST vom Typ LOGICAL*4; DUMMY ist eine beliebige Variable, sie wird bei diesem Aufruf nicht verändert).

```
CALL    $STAT1(LSTAT)
CALL    $STAT2(LSTAT,IFBER)
CALL    $STAT7(LSTAT,IFBER,IF1,IF2,IF3,IF4,IF5)
```

Bei jedem dieser Aufrufe werden der derzeitige Stand der Indikatorparameter an das Benutzerprogramm übergeben und die abgefragten Indikator-Parameter intern wieder auf ihre Anfangswerte zurückgesetzt.

Wenn der Steuer-Parameter NFALBE größer als 10000 gesetzt wird, nimmt das System an, daß der Statusindikator LSTAT niemals abgefragt wird und setzt ihn daher nicht mehr auf .FALSE.
Bei Trennung von Ausgabe und Fehlernachrichten kann es manchmal nützlich sein, eine eigene Nachricht zu schreiben (z.B. um die Stelle im Programm zu definieren). Dafür ist die Subroutine $MESS(FORMAT, VALUES,I) vorhanden. $MESS schreibt I Werte, die im Feld VALUES übergeben werden, mit dem im Feld FORMAT enthaltenen Format aus. Vor dem Schreiben wird jedoch das angelieferte Format etwas verändert und zwar werden die ersten 20 Zeichen des Feldes FORMAT mit einem Standardformatbeginn gefüllt. Dieser Standardformatbeginn enthält die Anfangsklammer des Formats, Vorschubzeichen sowie den Beginn des Nachrichtentextes. Von der 21.Stelle an muß also im Feld FORMAT dieser Standardtext mit dem individuellen Nachrichtentext fortgesetzt werden. Die ersten 20 Stellen des Feldes FORMAT, die in $MESS überspeichert werden, sind sinnvollerweise mit Blanks zu initialisieren. Der Rest des Formats darf nicht mehr als 117 Zeichen Ausgabe auf der ersten Zeile bewirken. Folgezeilen sollten der Übersichtlichkeit halber nicht vorgesehen sein.
Beispiel:

```
CALL $MESS('bb....(20Blanks)...bbENDE STEP1,A=",G12.4)',A,1)
```

Es sei ausdrücklich vermerkt, daß die führenden 20 freien Plätze im Feld FORMAT nicht mit dem Formatelement 20X erreicht werden können. Außerdem sei darauf hingewiesen, daß dort, wo innerhalb des Formats ein Apostroph erscheinen soll, zwei Apostrophe zu schreiben sind.

2.1.3 Aufruf von Sonderbibliotheken

In 2.1.1 wurde zusätzlich zu den bereits genannten Voraus - setzungen angenommen, daß

- das Standardzeichen $ in den Namen der Masterfunktionen nicht durch ein beliebiges anderes Zeichen z ersetzt wurde und

- die Datenfunktion nicht mit einem Synonym-Namen sname in MAPLIB integriert wurde.

Wären diese zusätzlichen Voraussetzungen in einer privaten Sonderbibliothek nicht gegeben, so müßten die Funktionen folgendermaßen aufgerufen werden:

$$x = zzzzzz(e,s,p_1,p_2,\ldots,p_n)$$

$$x = zzSTOF(e,p_1,p_2,\ldots,p_n)$$

$$x = EGzzzz(s,p_1,p_2,\ldots p_n)$$

$$x = sname(p_1,p_2,\ldots,p_n)$$

Zu beachten ist hier, daß auch bei Verwendung von Synonymen in e und s die Eigenschafts- und Stoffsymbole EG und STOF enthalten sein müssen.
Durch Abänderung des Zeichens $ in irgend ein anderes Zeichen z werden jedoch nur die Namen der Masterfunktionen verändert. Die Namen der MAPLIB-System-Routinen wie z.B. $STOP, $MESS, $ERROR enthalten nach wie vor das $-Zeichen.

2.1.4 Einbau der Routine in das Benutzerprogramm

Die in MAPLIB integrierten Routinen sind als Load Modules auf dem portitioned Data Set mit DSNAME=MAPLIB.IRE.LOAD abgelegt und können von dort über den Automatic Library Call des Linkage Editor oder Loader in das Benutzerprogramm eingebaut werden. Hierzu müssen die SYSLIB-Karten der Prozeduren mit folgenden Karten überschrieben werden:

```
//L.SYSLIBbDDb
//bDDb
//bDDbDSN=MAPLIB.IRE.LØAD,UNIT=2314,VØL=SER=GFK006,DISP=SHR
```

Werden die Referenzen von Loader gelöst, so steht auf der ersten Karte G statt L. Sollen noch aus weiteren Data-Sets Moduls geholt werden - z.B. aus der LØAD.IRE - so sind entsprechend weitere DD-Karten anzuschließen. Außerdem gibt es die Möglichkeit, sich von dem MAPLIB-Utility-Programm die erforderlichen Quellprogramme ausstanzen oder auf ein privates Magnetband schreiben zu lassen. Der Benutzer muß dann diese Programme gemeinsam mit seinem Problemprogramm compilieren.

Beispiel 1: Problemprogramm und MAPLIB-Routinen befinden sich auf Karten:

```
//C.SYSIN DD*
  Problemprogramm - Kartenpaket
  -
  MAPLIB-Routinen-Kartenpaket
  -
```

Beispiel 2: Problemprogramm befindet sich auf Karten;
MAPLIB-Routinen auf dem Band DV**** mit dem DSNAME name

```
//C.SYSINbDDb*
  Problemprogramm-Fortrankarten
/*
//bDDbUNIT=TAPE9,VOL=SER=DV****,DSN=name,DISP=(OLD,KEEP),
//bDCB=(BLKSIZE=800,LRECL=80,RECFM=FB)
```

Schließlich besteht die Möglichkeit die vom MAPLIB-Utility-Programm ausgegebenen Fortran-Karten compilieren zu lassen

und die compilierten Programme (Objektprogramme) über L.SYSLIN oder L.SYSIN dem Linkage-Editor bzw. über G.SYSLIN dem Loader zuzführen:

```
//G.SYSLINbDD
//bDDbUNIT=TAPE9,VOL=SER=DV****,DSN=name,DISP=(OLD,KEEP)
```

2.2 Benutzungsanleitung des MAPLIB-UTILITY-Programms

2.2.1 Einleitung

MAPLIB-UTILITY ist ein FORTRAN-Programm zur Verwaltung der MAPLIB-Daten=und Kontrollroutinen ("Routine" meint Function oder Subroutine).
Mit diesem Programm ist es möglich:

- Übersichten über die integrierten Routinen ausdrucken zu lassen
- die Routinen auf Papier oder andere Datenträger in verschiedenen Formen auszugeben
- neue Routinen zu vervollständigen und in MAPLIB zu integrieren
- Datenfunktionen mit Synonym-Namen in MAPLIB zu integrieren.

Da das Programm sehr umfangreich und jung ist, werden kleinere Änderungen in der Zukunft nicht vermeidbar sein. Der Benutzer wird von Änderungen vom Programm selbst durch entsprechende Meldungen am Anfang oder über die Subroutinen $NEWS, $STØP bei Verwendung der Datenfunktionen unterrichtet (siehe 2.1).

2.2.2 Schlüsselwörter und Codierungsregeln

Die verschiedenen Fähigkeiten des Programms werden über die Eingabe mit Schlüsselwörtern angesprochen. Eventuell erforderlich werdende Daten folgen in der Eingabe unmittelbar auf die entsprechenden Schlüsselwörter und werden durch das spezielle

Schlüsselwort "ENDb" abgeschlossen. Es gibt Haupt- und Folgeschlüsselwörter. Im Gegensatz zu Hauptschlüsselwörtern beginnen Folgeschlüsselwörter mit einem Punkt.
Folgeschlüsselwörter werden nur nach einem Hauptschlüsselwort verstanden.
Jedes Schlüsselwort muß einzeln auf einer Karte stehen und in Spalte 1 beginnen.
Haupt- und Folgeschlüsselwörter mit nur einem Punkt dürfen auf die vier ersten Zeichen abgekürzt werden, da nur diese unterschieden werden. Bei Schlüsselwörtern mit 2 Punkten werden 8 Zeichen abgefragt. Nach Schlüsselwörtern darf ab Spalte 5 bzw. 9 beliebiger Kommentar stehen.
Schlüsselwörter mit 2 Punkten beziehen sich auf eine bestimmte Routine; die 6 Zeichen nach den beiden Punkten entsprechen dem Namen der Routine.
Mit Ausnahme der Schlüsselwörter "NEW" und "OLD" können alle Schlüsselwörter in beliebiger Reihenfolge und beliebig oft aufeinanderfolgen.
Die beiden Ausnahmen sind nur in Sonderfällen notwendig; siehe unten.

2.2.3 Die Ein-/Ausgabeeinheiten

Das Programm benötigt eine Reihe von Ein-Ausgabeeinheiten, die in dieser Beschreibung mit symbolischen Namen bezeichnet werden. Für jede dieser Einheiten sind Standard-Fortran-Filenummern vorgesehen, die jedoch teilweise vom Benutzer mit dem Schlüsselwort UNIT verändert werden können (siehe unten).

Symbol	Standard File-Nr	Inhalt der Datei, die über diese Einheit angesprochen wird
INPUT	5	Eingabe der Schlüsselwörter und Daten
PRINT	6	Drucker-Ausgabe
PUNCH	7	Ausgabe von Routinen (z.B.über den Kartenstanzer
DIRECT	9	Zwischen-Speicher für Direct Access
SOURCE	11	Datei,d.sämtl.Fortran-Routinen enthält
REGISTER	13	Backup d.im COMMON enthaltenen Teilinformationen (vergl.3.2.2.2)
ERROR	6	Ausgabe der Kontrolle u.Fehlermeldungen

Stets erforderlich sind nur die Einheiten INPUT, PRINT sowie ERROR und REGISTER, die anderen nur bei entsprechender Eingabe. Anhang 3 zeigt eine Liste mit geeigneten OS-360/65-Job-Control-Karten, auf denen alle Dateien spezifiziert sind. Auf den Kommentarkarten ist angegeben, wann die verschiedenen Einheiten benötigt werden. Im Zweifelsfall können alle Dateien spezifiziert werden.

2.2.4 Übersicht über die Hauptschlüsselwörter und die durch sie angesprochenen Operationen

Schlüsselwort	Kurzerklärung
INFORMATION	Ausgabe von Information
SOURCE	Ausgabe von Routinen
LINK	Ausgabe von Routinen inclusive der zur Erzeugung von LOAD-Modules erforderlichen Job Control Statements.
ADD	Addieren oder Ersetzen von Routinen
COMMENT	Addieren oder Ersetzen von Information über die Stoff-, Eigenschafts- oder Parametersymbole
STORE	Abspeicherung und Fixierung der vorher eingegebenen Erweiterungen oder Änderungen
DELETE	Elimination von Routinen
NAME	Spezifikation des Füllzeichens der Namen der Masterfunktionen
UNIT	Änderung der Standard-File-Nummern der Ein-und Ausgabeeinheiten
COPY	Kopieren von Data-Sets von einer Einheit auf eine andere; zur Erzeugung von Backups
NEW	Information an das Programm, daß noch keine Datenbestände existieren.(Warnung! Falls doch schon Datenbestände existieren, werden diese durch NEW und nachfolgendes STORE zerstört)
OLD	Anweisung zur Übernahme der bisherigen Datenbestände. (Wenn weder OLD noch NEW angegeben ist, wird OLD angenommen; dies ist der Normalfall)

Man kann diese Hauptschlüsselwörter einteilen in die

- Ausgabeanweisungen SOURCE, INFO, LINK
- Eingabeanweisungen ADD, COMMENT, STORE
- Anweisungen zur Erzeugung von Teilbibliotheken DELETE u. NAME
- Sonderanweisungen NEW, OLD, UNIT, COPY

Entsprechend dieser Einteilung werden die Schlüsselwörter im folgenden zusammen mit ihren Unterschlüsselwörtern genauer beschrieben. Die einzelnen Gruppen sind soweit voneinander unabhängig, daß nur die interessierenden Abschnitte gelesen werden müssen.

2.2.5 Die Ausgabeanweisungen

Für einen ersten Überblick über die in MAPLIB integrierten Routinen dient die Anweisung

```
INFORMATION
.MAP
END
```

Um unnötige Produktion großer Papier- und Kartenmengen zu vermeiden, sollte man zuerst mit dieser Eingabe einen Utility-Job starten. Für die Erläuterung dieser Eingabe sei auf die folgenden Abschnitte verwiesen.

2.2.5.1 SOURCE

Das Hauptschlüsselwort SOURCE gibt an, daß Routinen über die PUNCH und gleichzeitig über die PRINT-Einheit ausgegeben werden sollen. Mit den nachstehenden Folgeschlüsselwörtern ist zu spezifizieren, welche Routinen ausgegeben werden sollen. Zumindest ein Folgeschlüsselwort muß angegeben sein, sonst wird nichts ausgegeben.

Folgeschlüsselwort	Auszugebende Routinen
.FUNCTIØN	alle Daten-Funktionen
.SYSTEM	alle System-Routinen
.MASTER	alle Master-Funktionen
..$$$$$$	nur die oberste Masterfunktion *)
..$$mmmm	nur die Stoff-Masterfunktion für *) den Stoff mit dem Symbol mmmm
..ee$$$$	nur die Eigenschafts-Masterfunktion für die Eigenschaft mit dem Symbol ee*)
..******	nur die spezielle Daten-Funktion oder Systemroutine mit dem Namen ******
. LIST	alle integrierten Routinen
.ALL	(ALL ist hier gleichbedeutend mit LIST, nicht aber nach dem Hauptschlüsselwort INFØ)

Für Sonderzwecke existieren außerdem die folgenden Schlüsselwörter:

.NEW

Die Ausgabe gemäß den Schlüsselworten

.FUN

.SYS

.MASTER

.LIST

.ALL

kann auf die neu eingebauten Routinen eingeschränkt werden,

---

*) Hier wird vorausgesetzt, daß das Standard-Zeichen $ nicht durch ein anderes mit dem Schlüsselwort NAME ersetzt wurde. Andernfalls ist an die Stelle von $ das neue Füllzeichen zu setzen.

die im selben Jobstep seit Beginn eingegeben wurden, wenn vor ihnen einmal das Schlüsselwort .NEW angegeben wird.

NOCOMMENT

Da die Fortran-Programme sehr viele Kommentarkarten enthalten, die für manche Anwendungen nur Ballast sind, gibt es die Möglichkeit, die Ausgabe der Kommentarkarten zu unterdrücken, indem das Schlüsselwort .NOC vor den,die Routinen spezifizierenden Schlüsselwörtern angegeben wird.

Beispiel:

```
SOUR
.SYS
..$$$$$$
..$$NA
..WLNAL
..EHNAV
.NOCOMMENT
.NEW
.FUN
END
```

Hiermit wird die Ausgabe bewirkt von:

- allen System-Routinen
- der obersten Master-Funktion
- der Stoff-Masterfunktion $$NA
- der Funktion WLNAL
- der Funktion EHNAV (bis hier mit Kommentarkarten)
- allen neu kurz zuvor in MAPLIB integrierten Daten-Funktionen ohne Kommentarkarten

Bei der Ausgabe erscheinen in Spalt 73-76 die ersten vier Zeichen des Routinen-Namens und in Spalte 77-80 eine laufende Kartennummer.

WARNUNG Bei Angabe der Schlüsselwörter .LIST oder .FUN oder .ALL erhält man sehr viel Papier und sehr viele Karten. Wenn nicht tatsächlich alle Routinen interessieren, sollte man daher unbedingt die Namen der gewünschten Routinen einzeln spezifizieren.

2.2.5.2 LINK

Das Hauptschlüsselwort LINK gibt an, daß Routinen über die PUNCH- und über die PRINT-Einheit ausgegeben werden sollen. Zur Spezifikation der Routinen, die ausgegeben werden sollen, dienen genau die gleichen Unterschlüsselwörter wie nach SOURCE. Zum Unterschied gegenüber SOURCE werden die Routinen auf der PUNCH-Einheit ausgegeben, einschließlich aller für die Erzeugung von Load-Moduls erforderlichen Job Control Statements. Für jede Routine wird ein Jobstep erzeugt. Sieben Job-Steps bilden einen Job mit eigener Job-Karte und Job-Ende-Karte. Wenn nichts anderes oder falsches angegeben ist, werden einige Parameter der Job-Control-Anweisungen standardmäßig eingesetzt. Jedoch können diese auch durch Eingabe von Daten nach den folgenden Schlüsselwörtern bestimmt werden:

Schlüsselwort: .JOB

Bedeutung:
Es folgt das zu verwendende Job-Statement auf einer oder auf zwei Karten. Um die Job-Karte(n) in der Eingabe als Daten zu kennzeichnen, sollten sie nicht mit Schrägstrichen(//), sondern mit zwei beliebigen anderen Zeichen beginnen. Bei der Ausgabe wird // automatisch eingesetzt.

Standardausgabe:
//IRE673nnbJOBb(0673,330,P0000),SCHUMANN,MSGLEVEL=(1,1),CLASS=A
nn nimmt fortlaufend die Werte 00, 01, 02,...,0Z,10....,ZZ an.

Schlüsselwort: .JIDENTIFIKATION

Bedeutung:

Die ersten beiden Zeichen auf der folgenden Karte (alphamerisch) sollen an der Stelle von nn auf der ersten Job-Karte stehen. Bei den folgenden Job-Karten wird wie oben weitergezählt.

Standardwert: 00

Schlüsselwort: .COMPILER

Bedeutung:

Die ersten vier Zeichen auf der folgenden Karte definieren den zu verwendenden Compiler.
Vorgesehen sind z.Zt.:

FGbb	FORTRAN-G-Compiler	
FH0b		OPT=0
FH1b	FORTRAN-H-Compiler mit	OPT=1
FH2b		OPT=2

Standardwert: FORTRAN-H-Compiler mit OPT=2

Schlüsselwort: .TIME

Bedeutung:

Im Format 2I1 ist auf der folgenden Karte die Zeit(in Sekunden) für einen Jobstep angegeben(zwischen 01 und 59 Sekunden)

Standardwert: 15
Diese Stepzeit, multipliziert mit 7, bestimmt die Job-Class

Schlüsselwort: .DSNAME

Bedeutung:

Die ersten vier Zeichen auf der folgenden Karte definieren den Ausgabe Data-Set, auf den die Load-Moduls vom Linkage-Editor geladen werden sollen. Vorgesehen sind:

LOAD.IRE sowie
MAPLIB.IRE.LOAD

Standardwert: LOAD.IRE

Beispiel:

```
LINK
.JOB
XXIRE999AAbJOBb(0999,0,0),IHRNAME,MSGLEVEL=(1,1),
XX     CLASS=B
END
.DSN
MAPL
END
.TIME
45
END
.JID
X1
END
.COM
FG
END
.NEW
.ALL
END
```

2.2.5.3 INFORMATION

Das Hauptschlüsselwort INFORMATION bewirkt die Ausgabe von Routinen, Listen und Tabellen lediglich auf der Ausgabeeinheit PRINT.

Welche Routinen ausgegeben werden sollen, wird mit den gleichen Unterschlüsselwörtern wie bei SOURCE angegeben (s.Kap.2.2.5.1)

Die Angabe folgender Schlüsselwörter ist hier zustätzlich möglich:

Schlüsselwort	Ausgabe
.MATERIAL	Liste der Materialsymbole mit Erläuterungen
.PROPERTY	Liste der Eigenschafts-Symbole mit Erläuterungen und Dimensionsangaben
.PARAMETER	Liste der Parameter-Symbole mit Erläuterungen und Dimensionsangaben
.TABLE	Tabelle der integrierten Funktionen, gekennzeichnet durch ihre Parameterliste eventl. mit ihren Synonymnamen, sowie eine Liste der integrierten Systemroutinen und eine Angabe über die Zahl der Records aus der SOURCE-Datei. Einschränkung: Falls die Parameterliste mehr als 24 Zeichen umfaßt (incl.Klammern und Kommata), werden die Parameter-Symbole auf ihren ersten Buchstaben verkürzt; ist die Parameterliste auch dann noch länger als 24 Zeichen, so werden nur die ersten 24 Zeichen der verkürzten Parameterliste angegeben; es fehlt dann unter anderem die rechte Klammer; Auskunft über die vollständige Parameterliste erhält man durch Ausdrucken der ganzen Funktion(..name) Die Tabelle enthält zusätzlich Informationen über Parameter, für die bei fehlenden Angaben Standardwerte eingesetzt werden.
.MAP	bewirkt die gleiche Ausgabe wie die Folge .MAT .PRO .ARG .TAB
.ALL	bewirkt hier die gleiche Ausgabe wie die Folge .MAP .LIST

2.2.6 Die Eingabeanweisungen

2.2.6.1 ADD

Das Hauptschlüsselwort ADD zeigt an, daß im folgenden Routinen kommen, die in MAPLIB integriert werden sollen. Es wird unterschieden zwischen Daten-Funktionen und System-Routinen. Welche Routinenart folgt, wird durch folgende Folgeschlüsselwörter angegeben:

.FUNCTION : es folgen Daten-Funktionen

.SYSTEM : es folgen System-Routinen

Nach diesen Unterschlüsselwörtern müssen die aufzunehmenden Routinen (eventuell mit vorhergehendem PROLOG; siehe unten) folgen. Die Daten müssen mit dem Schlüsselwort END (ab Spalte 1 zu lochen!) enden. Die Folgeschlüsselwörter können beliebig oft und in beliebiger Reihenfolge auftreten.

Beispiel:

```
ADD
.FUN
  Prolog(eventl)
  Funktionen
END
.SYS
  Prolog
  Routinen
END
.FUN
  Funktionen
END
```

Die Eingabe nach .FUN

Das Programm, das nach dem Unterschlüsselwort .FUN aufgerufen wird, heißt ADDFUN. ADDFUN erwartet Fortrankarten mit einge-

schobenen Kommentarkarten. Folgende Kommentarkarten-Identifikationen werden insbesondere abgefragt und die folgende Information ausgewertet (siehe auch Anhang 2):

C$M Von Spalt 7 an steht die Erläuterung des Stoffsymbols, das im Funktionsnamen

C$P Von Spalt 7 an steht die Erläuterung des Eigenschaftssymbols, das im Funktionsnamen verwendet wird. Ab Spalte 61 steht die physikalische Einheit (maximal 12 Zeichen). Beginnt die physikalische Einheit in einer anderen als Spalte 61 und steht unmittelbar davor ein $, so tabelliert ADDFUN die Einheit selbst in Spalte 61.

CP Diese Karte muß immer paarweise erscheinen. Auf der ersten Karte des Paares steht ab Spalt 7 einer der Parameter-Namen, die von der Funktion verwendet werden. Anschließend kann der Gültigkeitsbereich dieses Parameters angegeben werden. Auf der zweiten Karte (ebenfalls mit CP beginnend) steht die zugehörige Erläuterung und die Dimensionsangabe, wie auf der C$P-Karte.

C$R Diese Karte gibt an, daß die Funktion eine ältere Funktion gleichen Namens - falls vorhanden - ersetzen soll.(REPLACE) Falls diese Funktion neu ist, wird sie dennoch aufgenommen. Falls dagegen diese Karte fehlt und schon früher eine Funktion gleichen Namens existierte, wird die neue Eingabe nicht registriert.

C$Z Diese Karte gibt an,daß ADDFUN in die Funktion die Kontrollroutine $NUMBR zur Überprüfung der Argumentenzahl einbauen soll. Eingebaut wird die Anweisung:
CALL$NUMBR(name,'name$_1$',NUMBR$(n),n,1,I, & s)
name ist der Name der Funktion oder das Synonym entsprechend dem Function-Statement.
name1 ist der Name der Funktion nach den MAPLIB-Namenskonventionen.
n ist die Zahl der Parameter. Falls als erster Parametername DUMMY verwendet wird, ist n = 0.
Hier wird vorausgesetzt, daß die Funktion genau n Parameter erwartet, also nicht etwa bei fehlenden Parametern Standardwerte einsetzt.
Stimmt die Zahl der angelieferten Parameter überein oder ist sie größer als n, so wird die Rechnung beim folgenden Statement fortgesetzt. I ist dann gleich 1. Ist die Zahl der Parameter dagegen kleiner als n, so wird im Fehlerbereich 1 oder 4 ein Fehler signalisiert und im Normalfall die Programmausführung abgebrochen. Wenn die Fehler-Steuer-Parameter (vergl. Kap.2.1.2) es zulassen, wird

name der Standardwert 1.zugewiesen und dann zum Statement-label s gesprungen, der vor einer RETURN-Anweisung steht.

C$G Diese Karte gibt an, daß ADDFUN in die Funktion die Kontrollroutine $RANGE zur Überwachung des Gültigkeitsbereichs der Parameter einbauen soll. Hierzu ist erforderlich, daß die entsprechende CP-Karte mit dem Namen des Parameters sich ebenfalls im Kartenpaket befindet. Auf der Karte, auf der ab Spalte 7 der Parametername steht, muß im Anschluß daran der Gültigkeitsbereich angegeben sein in der Form:

CP pname amin3 <= amin2 <= pname1 <= amax2 <= amax3

pname	zu überprüfendes Argument
pname1	ist in der Regel identisch mit pname; wenn jedoch zuvor der Wert von pname nach pname1 gespeichert wurde, können pname und pname1 verschieden sein
amin3	Grenzen des Wertes der Variablen pname1;
amax3	bei Überschreitung wird ein Fehler aus Bereich 3 signalisiert
amin2	entsprechend für Bereich 2
amax2	hier können beliebige arithmetische Ausdrücke stehen

Für jeden Parameter, für den der Gültigkeitsbereich auf der CP-Karte angegeben wurde, wird folgendes Statement eingebaut:

```
CALL $RANGE (name, 'name1','pname', amin3, amin2,pname1,
             amax2, amax3, 'dimension',&s)
```

Liegt der Wert von pname1 in dem zulässigen Bereich amin2 <= pname1 <= amax 2, so wird normal weitergerechnet. Liegt er im Bereich amin3 $\leq$ pname1 $\leq$ amax 3, so wird ein Fehler signalisiert aber normal weitergerechnet. Liegt er außerhalb dieses Bereiches, so wird je nach Stand der Kontroll-Parameter das Programm abgebrochen oder der Funktion name der Wert 1 zugewiesen und zu dem Returnstatement mit der Marke s gesprungen.

C$T Diese Karte steht nach der letzten Anweisung, die Fehlerkontrollen durchführt. Sie ist erforderlich im Zusammen-

hang mit C$B(siehe unten), wenn von ADDFUN die Statements eingebaut werden sollen, die das Überspringen der Fehlerkontrollen ermöglichen (siehe unten). Die Karten zwischen C$F und C$B bilden die"Testzone" der Funktion. Wenn in die Funktion keine Fehlerkontrollen eingebaut sind, da diese aufgrund der Anweisungen C$Z und C$G von ADDFUN eingebaut werden sollen, kann C$T fehlen.

C$B Diese Karte ist die Anweisung an ADDFUN,in die Funktion Statements einzubauen, die das Überspringen der "Testzone" (siehe C$T) durch entsprechendes Setzen des Steuer-Parameters TEST (vergl.2.1.2) ermöglichen. Hierbei werden in die Funktion eingebaut:

vor C$F:

CØMMØN/$TEST$/NOTEST

LOGICAL*1 NOTEST

IF(NOTEST)GØTØ s

C$F

nach C$T:

s.....

s ist ein von ADDFUN bestimmter Statement label
NOTEST wird in den Steuerprogrammen gleich .NOT. TEST gesetzt

C$Ø Diese Karte kann insbesondere vor einem Fortran-END-Statement stehen. ADDFUN nimmt dann an, daß die folgenden Karten noch zu der vorhergehenden Routine gehören und einen gemeinsamen Load-Module mit der vorhergehenden Routine bilden sollen.
Hier können z.B. Unterprogramme oder BLOCK DATA-Blöcke kommen, die nur in Verbindung mit der vorhergehenden Routine verwendet werden.

C$N In Spalte 7 und 8 steht eine Integerzahl n, die angibt, wieviel Parameter die Funktion mindestens erwartet. In der Regel ist die Karte C$N überflüssig. Wenn n kleiner ist als die Gesamtzahl, dann ermöglicht die Angabe von n die Kennzeichnung der anderen Parameter als nicht unbedingt erforderlich in den Informationsausgaben.

z.B.:

```
      FUNCTION ROUØ(TK,PORVOL)
    C$N 1
      P = 0
```

```
IF(NUMBR$(2).EQ.2) P = PORVOL
.
.
.
```

In der Tabelle der Funktionen nach den Schlüsselwörtern INFO/.TAB erscheint in der Parameterliste nach n Parametern ein Semikolon: (TK;PORVOL)

C$S Diese Karte gibt ab Spalte 7 einen Synonym-Namen an. Dieser Name muß nach den MAPLIB-Namensgebung-Konventionen gebildet sein. Er enthält also das Eigenschafts- und Stoffsymbol, mit dem diese Funktion identifiziert wird. Dieser Synonymname steht dann an den Stellen von name 1. Für Name steht der tatsächliche Funktionsname

Weiterhin überprüft ADDFUN, ob die folgenden Informationsidentifikationen vorhanden sind:

CA Autor

C
CN Name; wenn nur CN angegeben ist, wird dennoch
C

```
C
CN  ***MAPLIB***FUNCTION
C
```

zusätzlich eingesetzt.

CD Datum

CL Literatur

C$F Die eingebauten Anweisungen folgen unmittelbar auf diese Karte. Wenn diese Karte fehlt, folgen die eingebauten Anweisungen als erste Anweisung nach dem FUNCTION-Statement. Wenn die Karten CZ, CG und C$B fehlen, kann auch C$F fehlen.

C$F ist insbesondere dann erforderlich, wenn in der Funktion Statement-Functions definiert werden, die gemäß FORTRAN-Vorschriften vor der ersten ausführbaren Anweisung stehen müssen.

Prolog

Einige der Karten brauchen nicht in der Funktion angegeben zu sein, wenn sie stattdessen im Prolog stehen.

Den Prolog bilden alle die Karten, die vor dem ersten FUNCTION-Statement stehen.
Ein Prolog ist insbesondere dann sinnvoll, wenn zahlreiche Funktionen nachfolgen.
Folgende Identifikationen können im Prolog erscheinen:

CA
CD
CL
CP
C$M
C$P
C$Z
C$G
C$R
C$B

Im Prolog müssen auch die C$M und C$P-Karten genau wie sonst die CP-Karten paarweise auftreten. Jeweils auf der ersten Karte steht ab Spalte 7 das Symbol, das auf der zweiten erläutert wird.

Die so im Prolog definierten Informationen werden in die nachfolgenden Funktionen eingebaut, es sei denn, sie sind dort ebenfalls angegeben. Dies gilt für alle Funktionen bis zum nächsten END-Schlüsselwort; danach ist die Prologinformation vergessen. Lediglich die Information auf CP, CM, CP-Karte wird ein für alle Mal aufbewahrt. D.h., ein einmal erläutertes Stoff-, Eigenschafts- oder Parametersymbol braucht nie ein zweites Mal erläutert zu werden. Wenn es danach ein zweites Mal erläutert wird, so wird für die Zukunft die jüngere Information aufbewahrt.

Standardreaktionen

CP in Verbindung mit C$G-Karten

Wie oben dargelegt, sind in Verbindung mit der C$G-Karte CP-Karten erforderlich. Diese müssen paarweise auftreten. Auf der ersten Karte steht der Variablenname und der Gültigkeitsbereich, auf der zweiten die Erläuterung.
Falls die Erläuterung bereits früher angegen wurde, braucht auf der zweiten Karte die Erläuterung nicht noch einmal wiederholt werden. Es genügt, wenn in Spalt 7 ein '$'(Dollar) Zeichen

steht. ADDFUN setzt dann die früheren Erläuterungen selbständig ein.

Falls irgendwelche Information weder in der Funktion selbst, noch im Prolog, noch jemals früher (bei CP, CM, CP) angegeben wurde, treten folgende Standardreaktionen auf:

CA	es wird "CA N.N." eingesetzt
CD	es wird "CD Datum der Rechnung" eingesetzt
CL	es wird nichts eingesetzt
CN	siehe oben
C$M	es wird "C$M UNKNØWN" eingesetzt
C$P	es wird "C$P UNKNØWN ?? " eingesetzt
CP	es wird "CP Parametername CP UNKNØWN ??" eingesetzt
C$R	es wird angenommen, daß die Funktion neu ist
C$Z	es wird angenommen, daß die Kontrolle die Argumentenzahl eingebaut ist
C$G	es wird angenommen, daß die Kontrolle des Gültigkeitsbereiches eingebaut ist
C$N	es wird angenommen, daß alle Parameter erforderlich sind (Regelfall)
C$B	es wird angenommen, daß die Möglichkeit zum Überspringen der Testzone entweder schon eingebaut ist oder nicht eingebaut werden darf.
C$F	Es wird die erste Karte nach dem Function-Statement, die von Spalte 1 bis 6 nur aus Blanks besteht, als erste ausführbare Anweisung angesehen.
C$T	Wenn C$T fehlt, erfüllt C$F die Funktion von C$T
C$Ø	Nach jedem END-Statement wird ein neues Function - Statement erwartet, es sei denn, das Schlüsselwert END folgt.

Bei fehlenden Erläuterungen CM, CP, CP werden Fehlermeldungen ausgeschrieben. Bei Änderungen, wird auf die Änderungen explizit hingewiesen.

Einschränkung der FORTRAN-Regeln

Aus praktischen Gründen sind folgende Einschränkungen zu beachten:

1) Die Fortran-Schlüsselwörter FUNCTION, SUBROUTINE, ENTRY, END sind ohne eingeschriebene Blanks ab Spalte 7 zu lochen.

2) Wenn C$Z oder C$G angegeben ist, muß wenigstens ein RETURN-Statement (mit oder ohne label) ab Spalte 7 ohne Blanks gelocht sein.

3) Das Function-Statement darf aus nicht mehr als sechs Karten bestehen. Es darf nicht mehr als 48 Parameter enthalten.

4) Die Namen NOTEST und ØTESTØ in Verbindung mit der Anweisung C$B, die Namen ØNUMBR und NUMBRØ in Verbindung mit C$Z und der Name ØRANGE in Verbindung mit C$G sind verboten. Ebenso selbstverständlich alle Namen von MAPLIB-Funktionen für andere Zwecke als den Aufruf dieser Funktionen.

Die Eingabe nach .SYS

Nach .SYS folgen Subroutinen oder Functions. An Kommentarkarten werden lediglich überprüft:

C$REPLACE

C$ØMIT

mit den gleichen Bedeutungen und Wirkungen wie nach .FUN.
Als Prolog darf nur die Karte C$R erscheinen. Auch hier sind die Fortran-Schlüsselwörter FUNCTION, SUBROUTINE, ENTRY und END ohne eingeschobene Blanks ab Spalte 7 zu lochen.

2.2.6.2 CØMMENT

Nach dem Schlüsselwort CØMMENT werden Daten in der Form eines Prologs gemäß dem vorliegenden Kapitel erwartet.

Ausgewertet werden Kommentarkarten mit den Identifikationen
CP, CP, CM.
Mit diesem Schlüsselwort ist es möglich, neue Symbole zu definieren und die Erläuterung alter Symbole zu ändern.

2.2.6.3 STORE

Mit dem Schlüsselwort STORE wird bewirkt, daß alle bisher über ADD, CØMMENT, SELECT oder NEU ausgeführten Änderungen des Datenbestandes fixiert werden. D.h.in einem späteren Job oder Jobstep erscheint der Datenbestand von MAPLIB in dem geänderten Zustand.
Wird die Anweisung STORE nicht oder falsch gegeben oder trat vorher ein MAPLIB-Systemfehler auf, so ist in einem späteren Job der Zustand von MAPLIB so wie zu Beginn des aktuellen Jobs.Zur Sicherung des Datenbestandes (vergl.Kap.3.2.2)wird hier ein Passwort, bestehend aus 8 Zeichen, auf der Karte nach STORE ab Spalte 1 erwartet. Wenn ein falsches Passwort angegeben wird, unterbleibt die Wirkung von STORE.
Beispielsweise lautet das Passwort: ABCDEFGH; dann ist anzugeben:

```
STORE
ABCDEFGH
END
```

2.2.7 Anweisungen zur Erzeugung von Sonderbibliotheken

2.2.7.1 DELETE

DELETE dient zum Löschen von Routinen mit ihren Eigenschafts- und Stoffsymbolen und von Parametersymbolen. Mit diesem Schlüsselwort kann der Benutzer alle ihn nicht interessierenden Routinen aus dem MAPLIB-Bestand eliminieren und sich so eine speziell für ihn geeignete Teilbibliothek erstellen. Er muß hierbei jedoch beachten, ob Routinen, die ihn nicht unmittelbar interessieren, von anderen Routinen aufgerufen werden.

Nach DELETE können in beliebiger Reihenfolge die Folgeschlüsselwörter .SYSTEM, .FUNCTION und .PARAMETER folgen, je nachdem Systemroutinen, Datenfunktionen oder Parametersymbole mit ihren Erläuterungen aus den Katalogen entfernt werden sollen.

Nach den Folgeschlüsselwörtern muß eine Liste der zu löschenden Namen, jeweils einzeln auf einer Karte ab Spalte 1 gelocht, folgen. Diese Dateneingabe muß durch das Schlüsselwort END abgeschlossen werden. Beispiel:

```
DELETE
.FUN
ESHEV
VONAVS
END
.SYS
$NEWS
END
.PARAMETER
TK
PN
PORVOL
END
```

Aufgrund dieser Eingabe werden die Funktionen ESHEV, VONAVS, die Systemroutine $NEWS sowie die Parametersymbole TK, PN und PORVOL gelöscht.

Werden aufgrund der Eingabe nach .FUN alle zu einem Stoff oder zu einer Eigenschaft gehörenden Funktionen gelöscht, so wird auch dieser Stoff bzw.diese Eigenschaft aus den Katalogen mit den zugehörigen Masterfunktionen und zugehörigen Erläuterungen gelöscht. Nicht dagegen wird überprüft, ob eventuell damit ein Parameter ohne Referenz bleibt und gelöscht werden könnte; dies muß der Benutzer selbst überprüfen.

2.2.7.2 NAME

Mit dem Schlüsselwort NAME wird das Füllzeichen der Master - funktionen definiert. Wird dieses Zeichen nicht explizit definiert, so wird als Füllzeichen das '$'(Dollar)-Zeichen verwendet.

Nach dem Schlüsselwort NAME wird lediglich eine Datenkarte erwartet. Das erste Zeichen auf dieser Karte wird danach als Füllzeichen verwendet. Es muß ein Buchstabe sein, der verschieden ist von I, J, K, L, M, N.

2.2.8 Die Sonderanweisungen NEW, OLD, UNIT, COPY

2.2.8.1 NEW und OLD

Die Schlüsselwörter NEW und OLD schließen sich gegenseitig aus. Nur eines von beiden darf also in der Eingabe vorkommen.
Wenn es angegeben wird, dann muß es auf der ersten Eingabekarte und nur da angegeben werden.
Steht auf der ersten Karte weder OLD noch NEW, so bewirkt dies das gleiche, als wenn vor dieser Karte noch die Karte OLD angegeben wäre.
OLD bewirkt, daß von der Einheit REGISTER der bisherige Katalog eingelesen wird. Auf der Eunheit SOURCE werden die bisherigen eingegebenen Routinen erwartet.
NEW bewirkt, daß der Katalog als leer deklariert wird. Diese Anweisung ist also sinnvoll, entweder bei der Erstellung völlig neuer Datenbestände oder zum Löschen sämtlicher alter Bestände (gefährlich!).
Im Normalfall wird daher keines der beiden Schlüsselwörter angegeben.

2.2.8.2 UNIT

Durch die Anweisung

```
UNIT
.einheit
nnpass
END
```

kann der Benutzer die Standard-Ein- und Ausgabeeinheiten mit dem symbolischen Namen einheit (vergl.2.2.3) auf beliebige andere FORTRAN-Einheiten nn abändern (Ausnahme: einheit darf nicht DIRECT oder ERROR sein). Es wird überprüft, ob die neue Einheit nn mit denen von DIRECT, SØURCE oder REGISTER übereinstimmt. Da dies zur Zerstörung der Datenbestände führen kann, sind diese Einheiten nur erlaubt, wenn das richtige Passwort pass angegeben ist. Andernfalls ist das Passwort nicht erforderlich. Das Passwort ist von den bei STORE und COPY (siehe unten) erforderlichen Passwörtern verschieden. Es besteht hier nur aus vier Zeichen.

Nicht geändert werden können DIRECT und ERROR. Fehlermeldungen kommen also immer auf die Einheit 6. Dieses Schlüsselwort ist für Sonderfälle gedacht. So kann z.B. die Ausgabe nacheinander auf verschiedene Dateien erfolgen. Eine häufige Anwendung ist jedoch die Unterdrückung von Ausgaben durch Änderung der Standard-Ausgabe-Einheiten auf DUMMY-Data-Sets.
Z.B.sei folgende DD-Karte vorhanden:

```
//FT01F001   DD   DUMMY
```

Dann kann die Druckausgabe unterdrückt werden, wie es z.B. bei zweimaliger SOURCE-Ausgabe sinnvoll wäre:

```
SOURCE
.LIST          Ausgabe auf PUNCH und PRINT
END
UNIT
.PRINT
01
END
SOURCE
.LIST          Ausgabe nur auf PUNCH
UNIT
06
END
```

2.2.8.3 COPY

Mit dem Schlüsselwort COPY soll ein weiterer Beitrag zum Punkt Sicherheit geleistet werden. Hiermit ist es möglich, ganze Data-Sets auf andere Einheiten zu kopieren. Insbesondere kann damit ein Backup von den Einheiten SOURCE und REGISTER erstellt werden, um den MAPLIB-Bestand nach einer eventuellen Zerstörung wieder regenerieren zu können. Die Ein-Ausgabe-Data-Sets werden folgendermaßen spezifiziert:

```
COPY
.FROM
ii
.TO
OO pass
END
```

ii — Einheit, von der kopiert werden soll

OO — Einheit, auf die der Data-Set geschrieben werden soll (ii und oo sind im Format I2 anzugeben)

pass — Passwort,bestehend aus 4 Zeichen; pass muß nur angegeben sein, wenn OO eine der folgenden vier Einheiten spezifiziert:

SOURCE
REGISTER
DIRECT
INPUT.

pass — ist verschieden von den bei STORE und UNIT erforderlichen Passwörtern.

Für ii und oo müssen entsprechende DD-Karten vorhanden sein. Wenn das Schlüsselwort .FROM oder.TO fehlt, werden für ii und oo die folgenden Standardwerte angenommen:

ii : Fortran-Nummer der Einheit REGISTER

oo : Fortran-Nummer der Einheit PUNCH

2.3 Anforderungen an neu zu integrierende Routinen

2.3.1 Vorschriften

Datenfunktionen, die in MAPLIB integriert werden sollen, müssen folgende Vorschriften beachten.

1) Sie müssen sich an die Namensgebungsvorschriften halten (vergl.Kap.1.3.2 und Anhang 1)
Soweit bereits definiert, sind bestehende Symbole zu verwenden. Über die bisher definierten Symbole informiert das Utility-Programm mit der Eingabe

```
INFO
.MATERIALS
.PROPERTIES
END
```

Neue Symbole sind nach den Regeln des Anhang 1 möglichst in Anlehnung an die englischsprachige Bezeichnung zu wählen.

2) Sie müssen ihre Parameter sowie ihr Ergebnis in den Dimensionen des M.K.S.A.K. System erwarten bzw. zurückgeben (vergl.Kap.1.3.3).

3) Sie dürfen nicht selbsttätig Nachrichten schreiben oder das Programm abbrechen (kein STOP).

4) Sie müssen die richtige Anzahl der Parameter und die Parameterwerte prüfen und eventuelle Fehlermeldungen durch Aufruf der unten beschriebenen Systemroutinen an die Kontrollprogramme mitteilen.

5) Falls bei Überprüfung der Parameterliste ein Fehler festgestellt wird, so ist gem.Kap.2.1.2, Tab.1 entweder der Funktionswert oder der Standardwert 1 zurückzugeben. Wenn möglich sollten bereits früher definierte Parameter-Symbole verwendet werden (siehe Ausgabe des Utility Programms nach Eingabe

```
INFO
.PARAMETER
END)
```

7) Programme, die die häufig verwendeten Parameter Temperatur TK und Druck PNM2 erwarten, sollen diese als erste Parameter in der Reihenfolge (TK, PNM2) führen. Die Einhaltung dieser Reihenfolge vermeidet Fehlerquellen.

8) Fehlernachrichten, die an die Systemroutinen zu übergeben sind, sollten nicht länger sein als eine Zeile. In der Fehlernachricht sollen der Name der Funktionen und des fehlerhaften Arguments enthalten sein, um den Benutzer eine einfache Fehlerfeststellung zu ermöglichen.

9) Es kann sinnvoll sein, in schon vorhandene Funktionen ein ENTRY-Statement einzubauen, das Funktionsnamen und die Parameter der alten Version enthält. Dadurch können schon vorhandene Benutzerprogramme weiterhin über diesen Eingang die neue Routine benutzen. Die Parameterliste des Entry-Statements muß die gleiche Parameterzahl sowie die gleichen Parameternamen enthalten wie die des Functionsstatements, allerdings nicht notwendig in gleicher Reihenfolge. Auch werden von MAPLIB keine Entry-Namen registriert. Im allgemeinen ist von Entries abzuraten.

10) Die Funktion soll von den in Anhang 3 angegebenen Kommentarkarten begleitet sein. Für die genaue Form dieser Kommentarkarten vergl.Kap.2.2.8.1 .

2.3.2 Fehlerkontroll- und Hilfsroutinen

Dem System werden Fehler über die Subroutine $WARN mitgeteilt. Hierbei sind auch das Format und die Werte der Fehlermeldung anzugeben.Um dem Benutzer die Feststellung der Fehler und die Erzeugung der Fehlermeldungen zu erleichtern, kann er sich jedoch auch - soweit möglich - der weiteren Kontrollroutinen bedienen.

1) $WARN

Aufruf: CALL $WARN(TEXT,VALUES,N,I)
Mit dieser Routine wird dem System mitgeteilt, daß im Fehlerbereich I ein Fehler aufgetreten ist. Das System soll die N im Feld VALUES enthaltenen Werte (REAL, INTEGER oder Zeichen) nach dem im Feld TEXT enthaltenen Format ausdrucken. Das Feld TEXT muß ohne Anfangsklammern mit 20 Blanks beginnen und am Ende die Endklammer des Formats aufweisen. Die ersten 20 Zeichen

des Formats werden vom System eingesetzt. Der Rest des Formats darf nicht mehr als 117 Zeichen Ausgabe auf der ersten Zeile bewirken. Folgezeilen sollten der Übersichtlichkeit halber nicht verwendet werden.(Vergl.$MESS in Kap.2.1.2)
In $WARN wird je nach dem Stand der Kontrollparameter die Fehlermeldung ausgeschrieben und eventuell die Programmausführung abgebrochen.

2) $TEXT 2 und $TEXT 3

Die Routinen $TEXT2 und $TEXT3 enthalten Standardformate, mit denen die Routine $WARN aufgerufen wird. I = 2, wenn $TEXT3 und I=3, wenn $TEXT3 aufgerufen wird.

Aufruf: CALL $TEXTi(KPROG,KARG,AMIN,AMAX,ARG,KDIM)

i = I

KPROG: INTEGER KPROG(2); enthält linksbündig den Namen der aufrufenden Funktion als Zeichenkette bestehend aus 6 Zeichen

KARG: INTEGER KARG(2); enthält linksbündig den Namen des fehlerhaften Arguments als Zeichenkette bestehend aus 6 Zeichen

AMIN: REAL; untere Grenze des Gültigkeitsbereiches für das bezeichnete Argument

AMAX: REAL; obere Grenze des Gültigkeitsbereiches für das bezeichnete Argument

ARG: REAL; Wert des Arguments

KDIM: INTEGER KDIM(3); enthält die physikalische Einheit des Arguments als Zeichenkette aus 12 Zeichen

Beispiel:

```
TK  =  100.
CALL $TEXT2('RØUØbb','TKbbbb',373.,1274.,TK,
1                          'GRADbKbbbbbb')
```

Ausgegeben wird vom System dann folgender Text (auf einer Zeile):

WARNING. ARGUMENT TK = 100.ØUT ØF RANGE 373 TØ 1274 GRAD K
IN RØUØ , VALUE MAY BE UNRELIABLE

3) $WARN2 und $WARN3

Um für Standardargumente nicht unnötig oft Argumentennamen und -Dimension speichern zu müssen, kann die gleiche Ausgabe wie mit $TEXT 2/3 auch mit folgendem Aufruf erreicht werden:

CALL $WARNi(KPRØG,IARG , AMIN,AMAX,ARG)

KPROG, AMIN, AMAX, ARG: siehe oben

IARG: INTEGER; definiert den Namen und die zugehörige Dimension des Arguments:

IARG	KARG	KDIM
1	TK	GRAD K
2	PNM2	N/M**2
3	ZSEC	SEC
4	CØNCTR	MOL-PROZENT
5	PØRVOL	VOL-PROZENT

Die Routine kann auch noch für andere Kombinationen Name/Dimension erweitert werden.

4) RANGE und $RANGE

Wenn der Gültigkeitsbereich der Funktion eine einfach zu beschreibende Umgrenzung besitzt, kann der Benutzer die Überprüfung der Parameter den Routinen RANGE und $RANGE überlassen. Mit diesen Routinen wird die Kontrolle des Gültigkeitsbereichs denkbar einfach. Wenn gewünscht und möglich, kann unter den genannten Voraussetzungen die Kontrolle des Gültigkeitsbereiches dann sogar vom MAPLIB-Utility-Programm eingebaut werden. In diesem Fall wird die Subroutine $RANGE verwendet.

Aufruf:

```
      FUNCTION PROG(ARG)
      CALL RANGE(PROG,'PROGbb',IARG,AMIN2,AMAX2,AMIN3,AMAX3,ARG,&s)

s     RETURN

       bzw

      FUNCTION  PROG(ARG)
      INTEGER KPRØG(2)/'PRØGbb'/
      CALL $RANGE(PROG,KPROG,KARG,
     1AMIN3,AMIN2,ARG,AMAX2,AMAX3,KDIM,&s)

s     RETURN
```

PROG: REAL — Name der Funktion

KPRØG,KARG,ARG,IARG,KDIM — siehe oben

AMIN2, AMAX2 — Gültigkeitsbereich für den Parameter ARG. Bei Überschreiten wird ein Fehler aus Bereich 2 signalisiert

AMIN3, AMAX3 — Mathematisch oder physikalisch möglicher Bereich für den Parameter ARG. Bei Überschreiten wird ein Fehler aus Bereich 3 signalisiert;

s — PROG wird der Standardwert 1 zugewiesen; es wird sofort zu den Statement mit der Statement-Nummer s gesprungen

Fehlermeldungen werden über $WARN2/3 bzw.$TEXT2/3 an $WARN übergeben. Dort wird über die Fortsetzung der Programmausführung entschieden.

5) RAGVL1

RAGVL1 ist nur anwendbar bei einem Paramater. RAGVL1 ist eine Function. Sie erfüllt gleichzeitig zwei Aufgaben:

1) Sie überprüft, ob der Wert des Paramaters im Gültigkeitsbereich liegt (ähnlich wie RANGE).
2) Sie berechnet den Wert der Function nach einem Standardverfahren.

Aufruf:

```
FUNCTION        PROG (ARG)
REAL A ( IA)       / Werte /
PROG = RAGVL1(KPRØG,IARG,IFORM,IA,
       AMIN2,AMAX2,AMIN3,AMAX3,ARG,A)
RETURN
END
```

KPROG,PROG,IARG,AMIN2,AMIN3,AMAX2,AMAX3,ARG siehe oben

IFORM identifiziert des Standardrechenverfahren

A REAL A(IA), Vektor der Koeffizienten

Zur Zeit ist in RAGVL1 nur ein Standardrechenverfahren vorgesehen. Es ist identifiziert mit IFORM = 1 und ist ein Horner-Verfahren zur Berechnung des Polynoms:

$$RAGVL1 = A(1)+A(2)*ARG+A(3)*ARG**2+$$

$$.... +A(IA)*ARG**(IA-1)$$

Auf Wunsch können in RAGVL1 jedoch auch noch andere Standardverfahren eingebaut werden.

6) $NUMBR, NUMBR$

$NUMBR ist eine Subroutine, sie dient zur Kontrolle der Zahl der Argumente. Im Falle eines Fehlers signalisiert sie über $WARN in Fehlerbereich 1 oder 4 einen Fehler an das System, bewirkt entsprechende Fehlermeldungen und je nach Stand der Kontrollparameter den Abbruch der Programmausführung.
NUMBR$ ist eine Assembler-Funktion. Sie liefert für ein beliebiges DUMMY-Argument die Zahl der tatsächlich an die Funktion übergebenen Argumente.
Im Normalfall braucht der Benutzer diese Kontrollroutine nicht selbst einzubauen; dies kann das MAPLIB-Utility-Programm übernehmen. Wenn in der Funktion jedoch verschiedene Argumenten-

zahlen vorgesehen werden sollen, muß diese Subroutine von Hand eingebaut werden (siehe Beispiel 2).

Aufruf:

```
FUNCTION  PROG ( ... )

CALL $NUMBR(PRØG,KPROG,NUMBR$(D),JSOLL,M,I,&s)

s  RETURN
```

PROG: Name der Funktion

KPROG: INTEGER KPROG(2): Name der Funktion als Zeichenkette, bestehend aus 6 Zeichen und linksbündig gespeichert

D Dummy-Argument; wird nicht verändert; hier sollte die maximale Soll-Zahl der Argumente stehen.(Für den Grund vergl.Kap.3.1.2.1)

JSOLL INTEGER JSOLL(M)
JSOLL ist ein Feld. Es enthält alle zulässigen Argumentenzahlen in aufsteigender Reihenfolge.

M Zahl der zugelassenen Argumentenzahlen.

I Nach dem Rücksprung aus $NUMBR gibt I an, der wievielte Sollwert JSOLL(I) noch kleiner oder gleich der tatsächlichen Anzahl ist.

s Rücksprung-Statement-Nummer, wenn zu wenige Argumente angeliefert wurden

Falls nur eine bestimmte Sollzahl vorgesehen ist, kann an der Stelle von JSOLL dieser Sollwert als Konstante oder undimensionierte Variable stehen; an der Stelle von M ist der Wert 1 zu übergeben. Falls die Funktion PROG nur ein Dummy-Argument erwartet (Name des Arguments zur Steigerung der Transparaenz möglichst:"DUMMY")ist JSOLL(1)=0;M=1.
Im Falle eines Fehlers im Bereich 4 erhält PROG den Standardwert 1 zugewiesen.

Beispiele:

```
FUCTION  FT4988(/DUMMY/)
CALL $NUMBR(FT4988,'FT4988',NUMBR$(0),0,1,I,&99999)
```

```
99999 RETURN
      END
```

2) FUNCTION RØUØ (/TK/,/PORVOL/)

```
      INTEGER JSOLL(2)/1,2/
C$N   1
      CALL $NUMBR(RØUØ,'RØUØbb',NUMBR$(2),JSOLL,2,I,&10)
      X  =  0.
      IF(I.EQ.2) X = PORVOL
C     FALLS NUR EIN ARGUMENT ANGELIEFERT WURDE, WIRD FUER
C     PORVOL DER STANDARD-WERT 0.ANGENOMMEN
          .
          .
          .
10    RETURN
      END
```

3. Programmbeschreibung

Während bisher nur die Funktionen und die Anwendung der verschiedenen Programme beschrieben wurden, soll nun erläutert werden, wie die verschiedenen Fähigkeiten verwirklicht werden.

3.1 MAPLIB-Bibliothekssystem-Routinen

Abb.4 zeigt eine große Übersicht und Abb.5 eine detaillierte Darstellung des Zusammenspiels der verschiedenen Bibliotheksroutinen. Den vom Programmumfang und für den Benutzer wichtigsten Block bilden die Datenfunktionen. Für diese sei auf die Informationsausgabe des Utility-Programms und die dort angegebene Literatur verwiesen.
Der Kern der System-Routinen ist $WARN mit ihren Entries $CATAL $STOP und $MESS. Alle anderen Blöcke haben direkt oder indirekt Anschluß an diesen Kern.
In dieser Beschreibung soll nur auf diesen Kern, auf die Kontrolle der Anzahl der Argumente, sowie die Masterfunktionen eingegangen werden.
Für die genaue Erläuterung sei auf die Programmlisten mit ihren Kommentarkarten verwiesen.

3.1.1 Die Fehlerkontrollroutinen $WARN und $CATAL

$WARN und $CATAL werden von den Master- und Datenfunktionen bzw. ihren Hilfsroutinen aufgerufen, wenn ein Fehler entdeckt wurde. Hierbei wird der Fehlerbereich spezifiziert (bei $CATAL standardmäßig 5) und die auszudruckende Fehlermeldung übergeben.
In dem COMMON-Block $KØNT$ befinden sich Kontrollparameter. Hier werden die Ist- und Sollwerte der Fehler fixiert. Bei jedem Aufruf von $WARN oder $CATAL wird der Zähler (NFB) für den jeweiligen Fehlerbereich um eins erhöht. Es wird - wenn die Steuerparameter es zulassen - die Fehlermeldung auf der spezifizierten Einheit ausgeschrieben. Wenn die zulässige Zahl der Fehler nicht überschritten wurde, wird in das aufrufende Programm

zurückgesprungen; andernfalls wird das ENTRY $STOP angelaufen.

Wenn auf diese Weise oder durch direkten Aufruf in $STOP gesprungen wurde, werden die neuesten MAPLIB-Nachrichten von der Routine $NEWS sowie eine Fehlerstatistik ausgedruckt. Anschließend läuft das Programm auf STOP und gibt so die Kontrolle an das Operating System ab. Schließlich existiert zu dieser Routine noch das ENTRY $MESS. Dies bewirkt lediglich die Ausgabe einer Benutzer-Nachricht. $MESS kann von überallher aufgerufen werden.
Zu beachten ist noch, daß sowohl bei Aufruf von $MESS als auch bei Aufruf von $WARN die ersten fünf Wörter (20 Zeichen) des Formates TEXT überschrieben werden. Es wird hier entweder der Text "WARNING" oder "MESSAGE" eingesetzt, um Fehler und Benutzernachrichten unterscheiden zu können. $CATAL wird lediglich von der Masterfunktion über die Routinen ERR$$$, ERRX bzw. ERRX$$ aufgerufen; dort ist die Fehlernachricht so geschrieben, daß sie mit "***WARNING" beginnt. Die drei Sterne sollen das Fehlen der Datenfunktion als besonders schwerwiegenden Fehler kennzeichnen.

3.1.2 Die Kontrolle der Anzahl der Argumente

Die Überwachung der Anzahl der Argumente geschieht mit den Routinen NUMBR$, $NUMBR und dem COMMON-Block $NUMB$.
In Fortran selbst ist es nicht möglich, in einer Routine festzustellen, mit wievielen Argumenten sie aufgerufen wurde. Funktionell ist daher die Assembler-Funktion NUMBR$ der Kern dieser Überwachung.

3.1.2.1 Die Assembler-Function NUMBR$ *)

Das Programm NUMBR$ (DUMMY) ist als Function geschrieben und wird wie eine Fortran-Function aufgerufen. Der Funktionswert

---

*) NUMBR$ wurde von A.Pee, Institut für Reaktorentwicklung, Karlsruhe, erstellt.

ist die Anzahl der Argumente, die dem Programm, welches NUMBR$ aufruft, angeliefert wird. Das Argument von NUMBR$ ist ein DUMMY-Argument, an dem nichts geändert wird, so daß eine beliebige Programm-Variable eingesetzt werden könnte. Wenn jedoch in den Datenfunktionen die maximal erforderliche Parameterzahl an der Stelle des DUMMY-Arguments eingesetzt wird, so ist damit die Voraussetzung gegeben, alle Datenfunktionen in der vorliegenden Form auch auf anderen Rechnersystemen einzusetzen, für die diese Assemblerfunktion nicht existiert. NUMBR$ ist dann durch eine Fortran-Funktion zu ersetzen, die als Funktionswert den Argumentenwert liefert. Der Benutzer muß dann selbst sicherstellen, daß er die erforderliche Parameterzahl liefert. Grundsätzlich muß es jedoch auch auf jeder anderen Maschine möglich sein, in der jeweiligen maschinennahen Sprache eine Funktion zu erstellen, die analog zu NUMBR$ die Zahl der Argumente feststellt; deshalb ist die auf die IBM 360 und ihre Fortran-Compiler zugeschnittene Assemblerfunktion in Anhang 4 genauer erläutert.

3.1.2.2 Der COMMON-Block $NUMB$

Der Common-Block $NUMB$ hat folgenden Aufbau:

```
COMMON/$NUMB$ / J,LJ
LOGICAL*1 LJ
```

LJ ist durch Block-Data als .TRUE.initialisiert.

J enthält die Anzahl der Argumente

LJ ist .TRUE., wenn J unbestimmt ist

und.FALSE. wenn J bestimmt ist.

Dieser Common-Block ist erforderlich, da die Datenfunktionen sowohl direkt vom Benutzerprogramm als auch indirekt über die Masterfunktionen aufgerufen werden können. Da die Masterfunktionen vom Utility-Programm so erstellt werden, daß die Datenfunktion mit der maximal erwarteten Zahl der Argumente aufgerufen werden, können die Datenfunktionen in diesem Fall die

Zahl der Argumente, die vom Benutzerprogramm eigentlich angeliefert wurden, nicht mehr feststellen.
Mit Hilfe von NUMBR$ wird daher der Wert von J nur bestimmt, wenn LJ gleich .TRUE. ist. Der Vergleich des Istwertes J mit den Sollwerten geschieht (mit der Routine $NUMBR) in den Datenfunktionen. Beim Rücksprung in die Masterfunktion wird LJ wieder .TRUE. gesetzt.

3.1.2.3 Die Subroutine $NUMBR

Die Subroutine $NUMBR wurde erstellt, um die Kontrolle der Argumentenzahl in den Datenfunktionen so einfach wie möglich zu gestalten. Lediglich $NUMBR muß den COMMON-Block $NUMB$ enthalten. In den Datenfunktionen braucht er dafür nicht enthalten zu sein. Hier wird die Istzahl mit den Sollzahlen verglichen. Wurden zu viele Argumente angeliefert, wird ein Fehler in Bereich 1 signalisiert aber sonst, wenn die Kontrollparameter es zulassen, normal weitergerechnet. Werden zu wenig Argumente angeliefert, wird ein Fehler in Bereich 4 signalisiert, der Funktion wird, wenn die Zahl der zulässigen Fehler nicht überschritten ist, der Standardwert 1 zugewiesen und es wird zum RETURN-Statement der Datenfunktion gesprungen.

3.1.3 Die Masterfunktionen

Die Masterfunktionen sollen den flexiblen Zugriff zu den Datenfunktionen ermöglichen. Ihnen werden als zusätzliche Parameter zu den Datenfunktionsparametern die Eigenschafts- und Stoffsymbole übergeben. Die Masterfunktionen besitzen intern Tabellen mit den integrierten Symbolen. Durch entsprechenden Vergleich der Symbole wird die gewünschte Funktion herausgesucht und aufgerufen.
Die Masterfunktion der höchsten Ebene sucht aufgrund des Stoffsymbols zunächst die gewünschte Masterfunktion der Stoffebene

heraus, die dann ihrerseits aufgrund des Eigenschaftssymbols die eigentliche Datenfunktion ermittelt.

Wenn das angegebene Symbol nicht in der Tabelle enthalten ist, wird im Bereich 5 ein Fehler signalisiert und normalerweise die Programmausführung abgebrochen. Wenn bei Aufruf der Masterfunktion der höchsten Ebene nicht wenigstens 2 Argumente übergeben wurden, wird im Bereich 4 ein Fehler signalisiert. Für die Fehlermeldungen werden die Funktionen

ERR$$$ (bei der obersten Masterfunktion)
ERRX (bei den Masterfunktionen der Eigenschaftsebene)
ERRX$$ (bei den Masterfunktionen der Stoffebene)

aufgerufen.

Diese Funktionen geben die Fehlermeldungen an $CATAL bzw.$WARN, setzen LJ .TRUE. (vergl.3.1.2)und liefern den Standardwert 1 zurück

3.2. Beschreibung des Utility-Programms

3.2.1 Allgemeines

Das MAPLIB-Utility-Programm ist das zentrale Verwaltungsprogramm der Bibliothek. Sein Zweck wurde in Kapitel 1 dargestellt. Die einzelnen Funktionen gehen aus der Benutzungsanleitung, Kap.2.2 hervor. Hier soll auf den internen Aufbau des Programms eingegangen werden.
Für dieses Programm wurden folgende Eigenschaften angestrebt:

- So weit wie möglich sollen alle Eingabefehler festgestellt werden.
- Die Fehlermeldungen sollen ausführlich und selbsterklärend sein.
- Das Programm soll auf Fehler "joberhaltend" reagieren, jedoch gleichzeitig den bisherigen Datenbestand sichern.

- Die erforderliche Eingabe wird auf das unbedingt erforderliche Minimum reduziert. Bei fehlenden Angaben werden Standardwerte angenommen.

- Soweit möglich ist die Eingabe lesbar; sie wird ausführlich dokumentiert.

- Das Programm ist modulartig aufgebaut; die verschiedenen Operatoren sind unabhängig voneinander in beliebiger Reihenfolge anwendbar.

- Da das Programm sehr viel Ein- und Ausgabe bewirkt, wurde zu Lasten der reinen Rechenzeit auf kurze Maschinenverweilzeit geachtet.

3.2.2 Datenfluß

3.2.2.1 Ein-/Ausgabe-Einheiten

Die im Programm vorgesehenen Ein- und Ausgabeeinheiten werden in diese Beschreibung mit symbolischen Namen bezeichnet, wie schon in Kap.2.2.3 angegeben.(Siehe Tabelle 4)

3.2.2.2 Logischer Datenfluß

Für die Beschreibung des logischen Datenflusses sind drei Ein-Ausgabeeinheiten sowie ein internes REGISTER mit einer Backup-Einheit maßgeblich:(vergl.Abb.7)

1) Ein-Ausgabe-Einheiten

   INPUT-Einheit: Eingabe neuer Routinen (und Steuerdaten)

   SØURCE-Einheit: Sammelstelle für die neue Eingabe und alle bisher eingegebenen Routinen.

   Output-Einheit: (z.B.PRINT-Einheit, PUNCH-Einheit)
   Ausgabe von Routinen vom SOURCE-FILE her.

2) Register

   Logisch enthält die SOURCE-Einheit zwar vollständig die gesamte bislang in MAPLIB eingegebene Information. Für einen hinreichend schnellen Zugriff zu den Informationen werden jedoch charakteristische Teilinformationen gesondert abgespeichert.

Tabelle 4

Die Dateien des MAPLIB-Utility-Programms

Symb. Datei Name	Stan-dard File Nr	Inhalt	Datei-Typ
INPUT	5	Eingabe der Schlüssel-wörter und Daten	Sequentiell;LRECL=80 (REWIND und BACKSPACE muß möglich sein)
PRINT	6	Utility-Drucker-Ausgabe	System-Druckausgabe LRECL = 133
ERROR	6	Utility-Kontroll- und -Fehlermeldungen	System Druckausgabe LRECL = 133
PUNCH	7	Utility-Stanz-Ausgabe	System-Stanzausgabe LRECL = 80
DIRECT	9	Zwischen-Speicher	Direktzugriff (200 Blöcke zu je 4000 Bytes)
SOURCE	11	Datei, die sämtliche FORTRAN-Routinen enthält	Sequentiell; Ein-/Aus-gabe geschieht format-frei; LRECL=80
REGISTER	13	Backup der im COMMON enthaltenen Teilin-formationen (vergl. 3.2.2.2)	Sequentiell; Ein-/Aus-gabe unter Format-Kontrolle; LRECL=80 (LRECL=Logische Satz-länge)

Diese Teilinformation wird während der Ausführung von MAPLIB-Utility in COMMON-Bereichen des Programms abgespeichert. Für die Aufbewahrung zwischen den einzelnen Utility-Ausführungen wird eine Kopie der COMMON-Bereiche auf einem speziellen DATA-Set aufbewahrt, zu dem das Programm über die REGISTER-Einheit Zugriff hat.

Anhang 5 zeigt die Struktur der Teilinformationen.

3.2.2.3 Praktischer Datenzugriff

Der Zugriff zu den Daten geschieht, je nach der Anzahl und Sequenz der zu lesenden Karten, entweder auf den Einheiten selbst durch sequentielles Lesen oder nach Umspeicherung auf die DIRECT-Einheit im Lesen mit Direct-Access. Auch beim Arbeiten mit Direct-Access wird nicht direkt, sondern mit Hilfe eines programmeigenen Buffers gelesen (in der Routine GETREC).

Sollen weniger Karten im Direktzugriff verarbeitet werden, als auf einmal in diesem Buffer Platz haben, so unterbleibt die Umspeicherung auf die DIRECT-Einheit (vergl. die Liste der Routine FILFIL).

Diese verschiedenen Möglichkeiten sind eingebaut, um nicht nur die reine Rechenzeit, sondern auch die Maschinenzeit des Programms möglichst klein zu halten.

3.2.2.4 Sicherung des Datenbestandes

In dem Programm wurde versucht, eine Zerstörung des bisherigen Bestandes auf der SOURCE-Einheit durch fehlerhafte Eingabe oder Programmfehler weitgehend unmöglich zu machen.
Diese Datensicherung wurde mit folgenden Mitteln erreicht:

1) Jedes Schreiben auf die SOURCE-Einheit geschieht zunächst nur auf den Bereich nach dem letzten, in früheren Jobs geschriebenen Record. D.h. bei Änderungen alter Routinen, bleibt die alte Routine zunächst physikalisch unverändert. Während des aktuellen Jobs kann jedoch nicht mehr zu der alten Routine zugegriffen werden, da die entsprechenden Record-Nummern im COMMON geändert werden. Die ursprünglichen Record-Nummern stehen jedoch vorläufig unverändert auf der Register-Einheit.

2) Die Fixierung des geänderten Datenbestandes geschieht mit dem Befehl STORE, der das Comprimieren der Daten auf der SOURCE-Einheit sowie ein Updating der REGISTER-Einheit bewirkt. Dieser Befehl wird nur ausgeführt, wenn

- kein interner Programmfehler auftrat (angezeigt durch einen internen Statusindikator)

- der Befehl von einer hierzu berechtigten und über die Folgen informierten Person gegeben wird; die Berechtigung ist durch die Angabe eines Passwortes nachzuweisen, das vom Systemverwalter jederzeit durch Änderung der Function PASSW geändert werden kann.

- Werden in der neuen Information formale oder Eingabefehler entdeckt, so wird die neue Information erst gar nicht in dem temporären COMMON-REGISTER katalogisiert.

3.2.2.5 Grenzwerte für die zu verarbeitenden Datenmengen

MAPLIB-Utility ist ein Fortran-Programm. Es enthält daher festdimensionierte Felder, deren Größe den möglichen Umfang der zu verarbeitenden Daten beschränkt. Es wurde darauf geachtet, daß derartige feste Grenzen nur bei COMMON-Variablen auftreten. Wenn die gewählten Grenzen nicht ausreichen, müssen nur die Common-Statements und der BLOCKDATA-Block geändert werden.
Tabelle 5 zeigt die derzeitigen Dimensionsgrenzen.
Zusätzlich besteht die nicht vom COMMON her bestimmte Einschränkung, daß die Parameterliste einer Funktion nicht mehr als 48 Parameter enthalten darf. Dieser Wert ist in den Routinen ADDFUN und FUNANA im Feld ARGAKT (3,48) fixiert.

Neben diesen Dimensionierungsbeschränkungen bestehen Grenzen bei den verwendeten Data-Sets. Insbesondere ist hier im Programm die Grenze der Direct-Access-Einheit maßgebend, die maximal 10000 Karten fassen kann. Da an verschiedenen Stellen der gesamte Datenbestand von der SOURCE-Einheit auf die DIRECT-Einheit geschrieben wird, darf auch diese Einheit nicht mehr als 10000 Karten enthalten.

Tabelle 5

Dimensionsgrenzen, die zu beachten sind

Inhalt der Felder	Maßgebende Variable	Maximal Wert	Felder
Datenfunktionen	JEMAX	500	STO(JEMAX), ARG(6,JEMAX), NRECA(JEMAX), NRECE(JEMAX), EIG(JEMAX), NARG(JEMAX), LSYN(JEMAX),
Synonyme	KYMAX	20	SYNSTØ(KYMAX), JSYN(KYMAX), SYNEIG(KYMAX)
Systemroutinen	K$MAX	50	NRECSA(K$MAX), NRECSE(K$MAX), SYSTEM(3,K$MAX),
Eigenschaften	KEMAX	100	ERKEIG(17,KEMAX) NREIGA(KEMAX) NREIGE(KEMAX) KATEIG(KEMAX)
STOFFE	KSMAX	100	ERKSTO(17,KSMAX) NRSTOA (KSMAX) NRSTOE(KSMAX) KATSTØ (KSMAX)
Parameter	KAMAX	50	ERKARG(17,KAMAX) KATARG(3,KAMAX)
Maximale Zahlen der Datenkarten, die als ein Block nach ADD ausgegeben werden können	-	4000	LERL (4000)

Mit Ausnahme der Grenze 4000 des Feldes LERL werden im Programm alle Grenzen bei Erweiterungen des Datenbestandes abgeprüft. Bei Überschreitung der Grenzen wird der Indikator für Systemfehler "ERROR" gleich .TRUE. gesetzt. Nur bei Überschreitung des Platzes von 10000 Karten auf der DIRECT-Einheit wird die Programmausführung nach einer entsprechenden Fehlermeldung sofort abgebrochen (STOP in GETREC). Sonst wird die Ausführung des aktuellen Eingabebefehls abgebrochen und beim nächsten Befehl die Programm-Ausführung fortgesetzt.

Alle hier aufgeführten Grenzen können bei Bedarf verändert werden. Der verfügbare Kernspeicher und Direct-Access-Platz an der Anlage IBM 360/65 in Karlsruhe ließe dies zu.

In dem gegenwärtigen Zustand benötigt das Programm zur Ausführungszeit nach Eröffnung aller Ein- und Ausgabeeinheiten ca.220K Kernspeicher-Bytes (1K = 1024).

3.2.3 Programmablauf

Der Programmablauf wird von außen durch die Eingabe gesteuert. Hierzu existiert eine Subroutine MAPLIB, die die eingegebenen Schlüsselworte liest und dementsprechende Ausführungen durch Aufruf von Unterprogrammen veranlaßt. Die verschiedenen Unterprogramme sind in drei Klassen einzuteilen:

- Steuerungs- und Ausführprogramme
- Ein-/Ausgabeprogramme
- Hilfsroutinen

Einen Überblick über die Zwecke der verschiedenen Steuerungs- und Ausführprogramme zeigt Abb.8.Der Übersichtlichkeit halber sind hier Routinen der beiden anderen Klassen nicht wiedergegeben. Für die Programmdetails sei auf die ausführlich mit erklärenden Kommentarkarten versehenen Fortranlisten verwiesen.

3.2.4 Besonderheiten und Kritik an FORTRAN

MAPLIB-UTILITY ist ein Programm in dem kaum arithmetische Ausdrücke vorkommen. Stattdessen ist es ausgezeichnet durch:

- viel Ein-/Ausgabe
- zahlreiche IF-Statements
- Zeichenverarbeitung

Insbesondere für die beiden letzten Punkte ist FORTRAN nicht prädestiniert.
IF-Statements werden unhandlich, weil FORTRAN keine Block-Struktur kennt.
Für Zeichenverarbeitung ist FORTRAN überhaupt nicht ausgelegt. Dies ist nur über recht umständliches Arbeiten mit Variablen vom Typ LOGICAL*1 (1Byte lange Variablen; 1 Byte kann gerade ein Zeichen aufnehmen) und Equivalence-Statements möglich. Beide Schwächen machen das Programm schlecht lesbar. Bezüglich der Textverarbeitung wird das Programm zudem maschinenabhängig, da es auf Maschinen ausgerichtet ist, bei denen die Wortlänge 4 Bytes beträgt.
Trotz dieser Nachteile wurde das Programm in FORTRAN und nicht etwa in PL/1 geschrieben, um auf möglichst vielen Maschinen einsetzbar zu sein.

L i t e r a t u r

/1/ Studie über Möglichkeiten und Grenzen einer umfassenden Werkstoffdatenbank, unveröffentlicht;
Battelle Institut, Frankfurt; im Auftrag des Institutes für Dokumentationswesen, Frankfurt (1970)
siehe auch:
Oberender,W.:
Möglichkeiten und Grenzen einer umfassenden Werkstoffdatenbank (WDB)
DIN-Mitteilungen 49 (1970), 7, S.258-260

/2/ Pee,A., Schumann,U.:
An Integrated Program Library for Material Property Data
Nucl.Eng.Desg. (erscheint 1970)

/3/ Hoare,C.A.R.:
Panel discussion in data structuring
in:
Buxton,J.N.(ed.): Simulation Programming Languages (Proceedings of the IFIP working Conference on Simulation Programming Languages). North-Holland Publ.Co., Amsterdam 1968, S.158-174

/4/ Deutscher Normenausschuß
DIN 1301: Einheiten
Beuth-Vertrieb, Mai 1969
International Organization for Standardization:
ISØ Recommendation R 1000
Rules for the Use of Units and a Selection of the Decimal Multiples and Sub-Multiples of the SI Units
1st Edition, Geneva, Febr. 1969

/5/ Deutscher Kältetechnischer Verein:
DKV Arbeitsblatt 0-05 und 0-06
Internationales und technisches Maßsystem

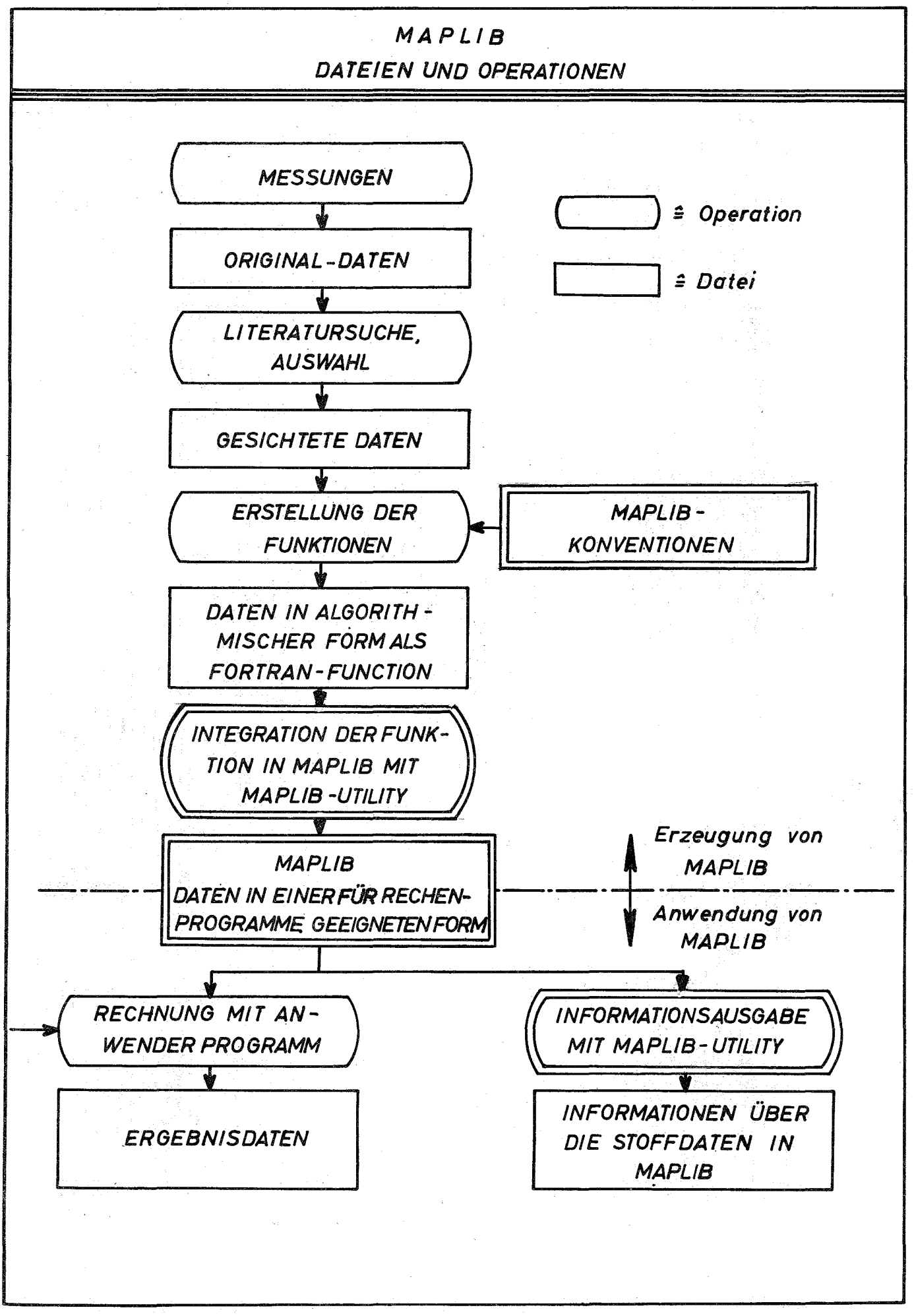


Abb. 1: Überblick über MAPLIB

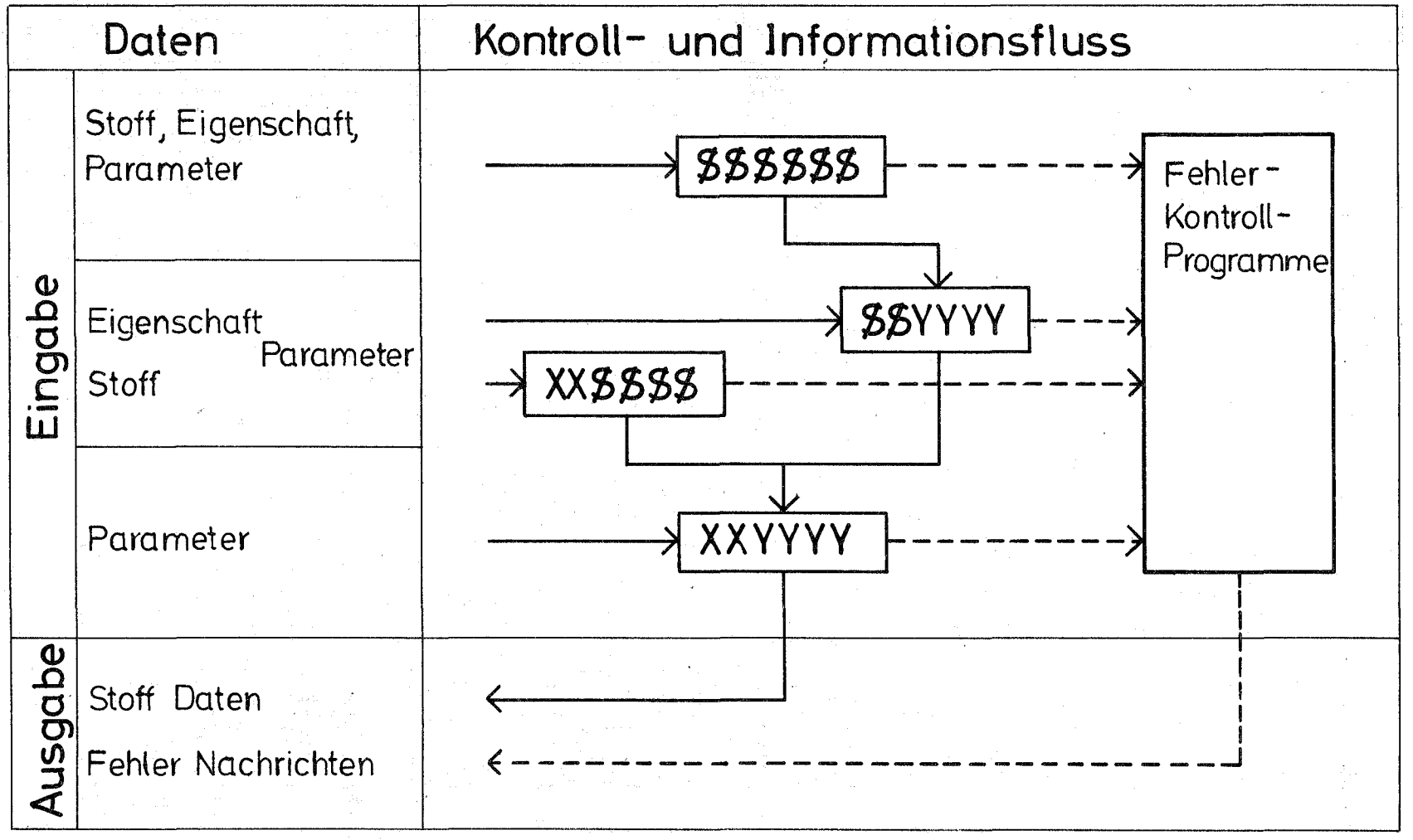


Abb.2: Zugriffssystem

```
                   $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
CARD     1         $      FUNCTION EHHEV (/TK     /,/PN     /)                 $
CARD     2         $C                                                          $
CARD     3         $CN    ***  M A P L I B  ***   FUNCTION                     $
CARD     4         $C                                                          $
CARD     5         $CD    06.05.70                                             $
CARD     6         $CA    ZIMMERER                                             $
CARD     7         $C$P   ENTHALPIE                               J/KG         $
CARD     8         $C$M   HELIUM-GAS                                           $
CARD     9         $CP    TK    10.<=273.<=TK<=1473.<=50000.                   $
CARD    10         $CP    TEMPERATUR                              GRAD K       $
CARD    11         $CP    PN    1.<=9.8067E4<=PN<=9.8067E6<=9.8067E7           $
CARD    12         $CP    DRUCK                                   N/M**2       $
CARD    13         $C$R                                                        $
CARD    14         $CL    K. BAMMERT,DIE THERMODYNAMISCHEN EIGENSCHAFTEN VON HELIUM ALS  $
CARD    15         $CL    ARBEITSMEDIUM FUER NUKLEARE GASTURBINEN              $
CARD    16         $CL    KERNTECHNIK,11(1969),88-91                           $
CARD    17         $C$F                                                        $
CARD    18         $      CALL $NUMBR(EHHEV ,'EHHEV ',NUMBR$( 2), 2,1,I,&  999)  $
CARD    19         $      CALL RANGE(EHHEV,'EHHEV ',1,273.,1473.,10.,50000.,TK,&999)  $
CARD    20         $      CALL RANGE(EHHEV,'EHHEV ',2,9.8067E4,9.8067E6,1.,9.8067E7,PN,&999)  $
CARD    21         $      EHHEV= 5196.*(TK-273.16)+3130.*(1/TK**0.3333-4715./TK**2)*1.E-5*PN  $
CARD    22         $     1-285.                                                $
CARD    23         $  999 RETURN                                               $
CARD    24         $      END                                                  $
                   $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

Abb.3: Beispiel für eine mit Kommentarkarten versehene Funktion

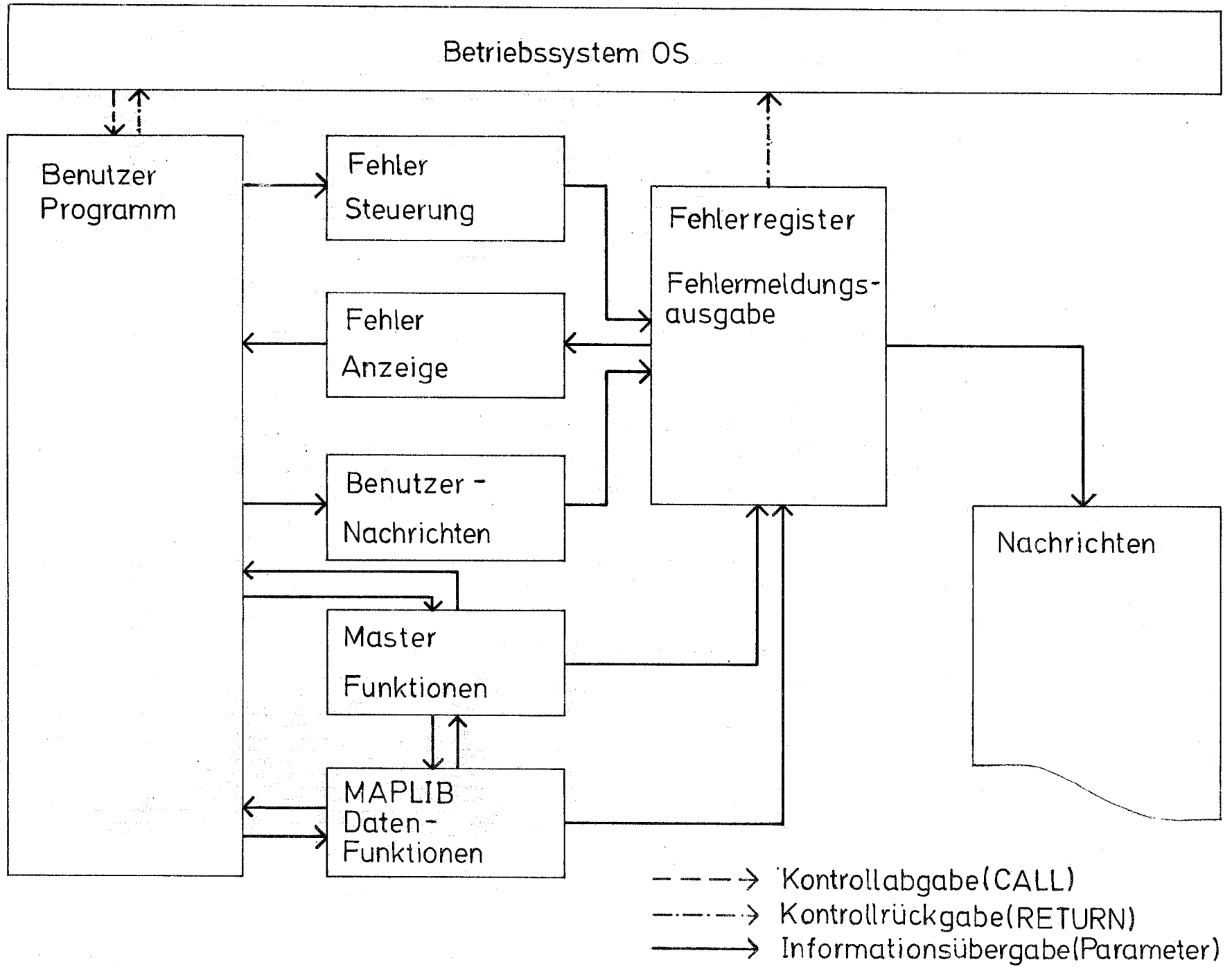


Abb.4: Übersicht über das MAPLIB-Programmsystem

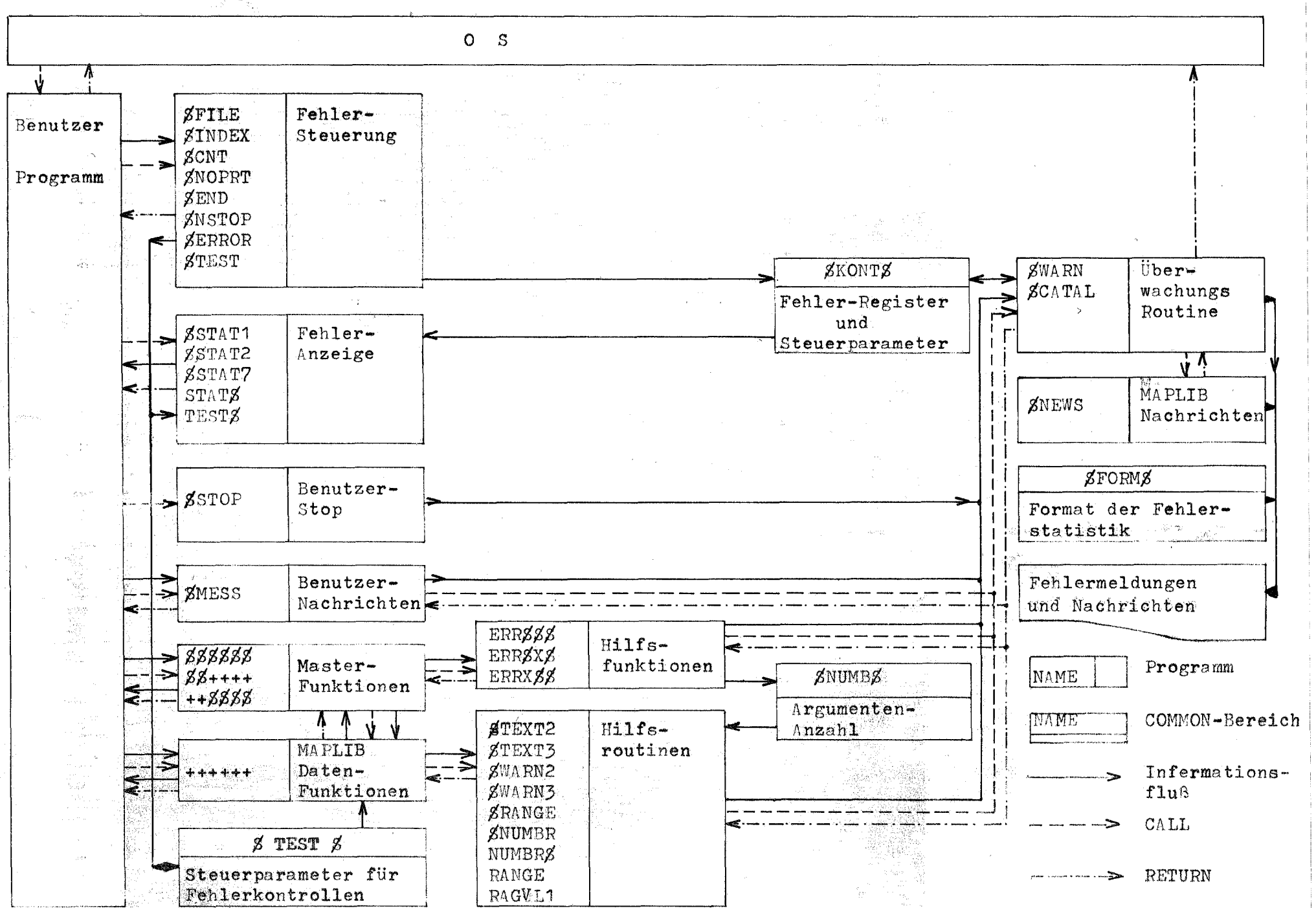


Abb. 5: Übersicht über die MAPLIB-Routinen

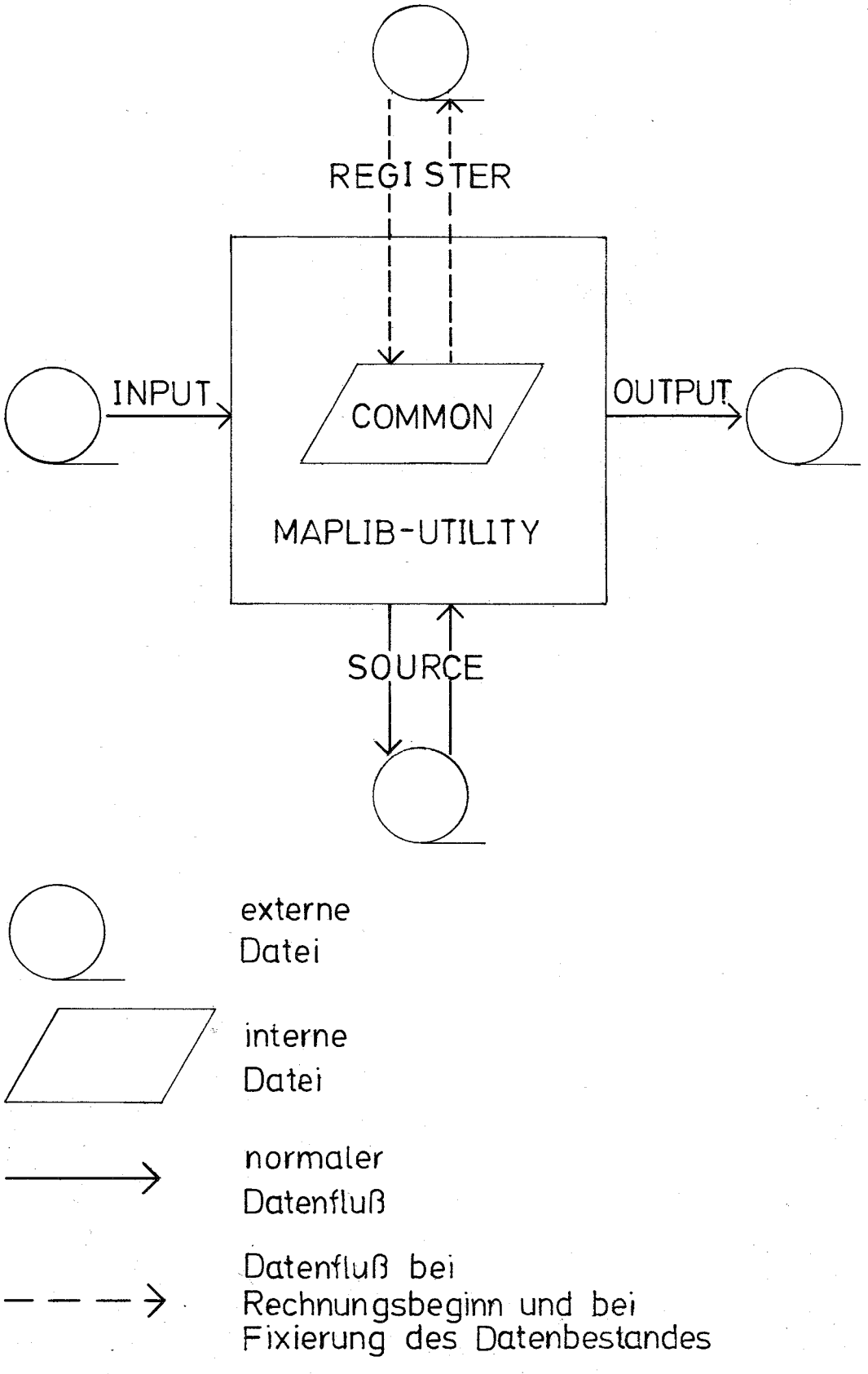


Abb.6: Logischer Datenfluß im Utility-Programm

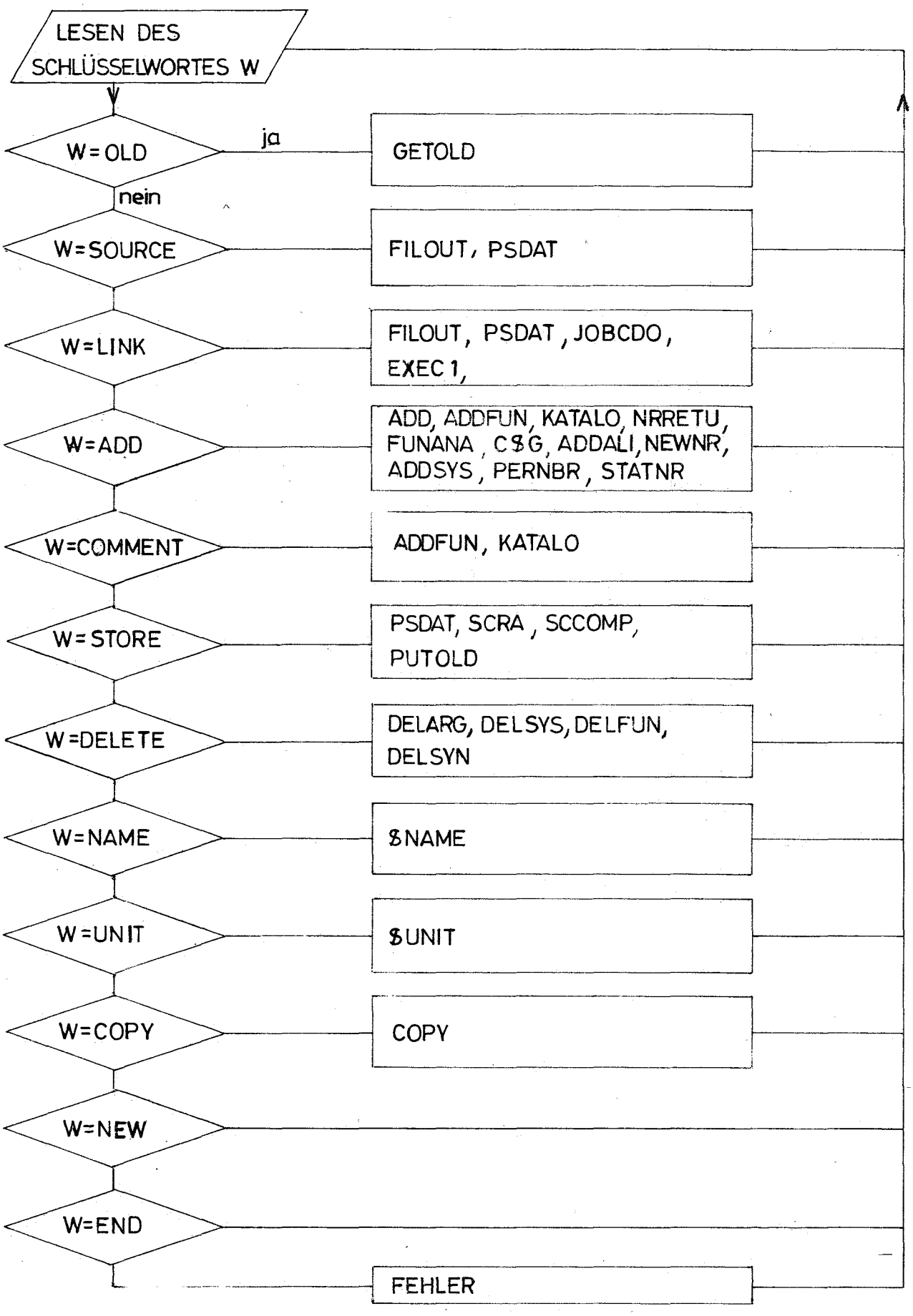


Abb.7: Überblick über die Utility-Routinen

A n h a n g 1

Namensgebungsregeln

Wenn für einen Stoff oder für eine Eigenschaft einmal ein Symbol festgelegt wurde, darf hierfür kein zweites Symbol definiert werden.

1. Stoffsymbole

- Stoffe werden mit mindestens einem und maximal vier alphamerischen Zeichen gekennzeichnet.

- Die vier Zeichen sollen nicht alle mit demselben Buchstaben gebildet sein, weil dann dieser Buchstabe nicht als Füllzeichen der Masterfunktion verwendet werden kann.

- Einige Regeln sollten soweit wie möglich eingehalten werden:

  1. Chemische Elemente werden mit ihren chemischen Zeichen definiert. Natrium = NA
  2. Einfache chemische Verbindungen werden durch ihre chemische Formel definiert. Wasser = H2O.
  3. Werkstoffe werden mit ihrer Werkstoffnummer definiert. Stahl 4981 = 4981
  4. Isotopen werden mit ihrem chemischen Zeichen, gefolgt von der vollständigen oder auf die wesentlichen Ziffern gekürzte Nukleonenzahl, definiert.
     Uran 238 = U238, Plutonium 239 = PU39
  5. Wenn die vier Zeichen nicht alle benötigt werden, können zusätzliche Angaben in das Symbol eingefügt werden, wie z.B.über die Phase.
     Natriumflüssig = NAL, Wasserdampf = H2OV,
     Natriumdampf an der Sättigungslinie = NALS.
  6. Wenn trotz dieser Regeln noch Freiheiten bestehen, sollte sich das Symbol an die englischsprachige Bezeichnung anlehnen. z.B. Luft = AIR

- Bisher wurden folgende Stoffsymbole festgelegt (nicht für alle aufgeführten Stoffe enthält MAPLIB bereits Datenfunktionen). Vergleiche hierzu auch die Informationsausgabe des MAPLIB-Utility-Programms

Symbol	Stoff
AIRV	Luft, gasförmig
B4C	Bor - 4 - Carbid
CO2V	Co_2 - Gas
FR2 2	Freon 22
H2O	Wasser, allgemein
H2OV	Wasserdampf
H2OL	Wasser, flüssig
K	Kalium,
KL	Kalium, flüssig
KV	Kaliumdampf
KLS	gesättigtes flüss.Kalium
KVS	gesättigter Kaliumdampf
NA	Natrium
NAL	flüssiges Natrium
NALS	gesättigtes flüssiges Natrium
NAV	Natriumdampf
NAVS	gesättigter Natriumdampf
PUC	Plutonium-Carbid
PUO	Plutonium-Oxid
UC	Uran-Carbid
UO	Uran-Oxid
UPUO	Uran-Plutonium-Mischoxid
U235	Uran 235
U238	Uran 238
PU39	Plutonium 239
4301	Stahl 4301
4401	" 4401
4550	" 4550
4961	" 4961
4981	" 4981
4988	" 4988
7380	" 7380

2. Eigenschaftssymbole

- Eigenschaften werden mit zwei alphamerischen Zeichen gekennzeichnet.

- Das erste Zeichen muß ein Zeichen aus der Menge (A,B,C,D,E, F,G,H,O,P,Q,R,S,T,U,V,W,X,Y,Z) sein.

- Das zweite Zeichen kann ein beliebiges alphamerisches Zeichen sein. Es soll vom ersten Zeichen verschieden sein, weil sonst dieses Zeichen nicht als Füllzeichen der Masterfunktionen verwendet werden kann.

- Einige Regeln sollten so weit wie möglich eingehalten werden:

  1) Funktionen, die Spannungen errechnen, führen als erstes Zeichen ein S.

  2) Funktionen, die Stoffeigenschaften für einen Phasenwechsel liefern, führen als erstes Zeichen die Bezeichnung des Phasenwechsels: F = Schmelzen, V = Verdampfung, R = Rekristallisation.

  3) Das Eigenschaftssymbol kann, wenn möglich und passend, dem für diese Eigenschaft in der technisch-wissenschaftlichen Literatur üblichem Symbol entsprechen: Spezifische Wärme bei konstantem Druck: CP

  4) Wenn trotz dieser Regeln noch Freiheiten bestehen, sollte sich das Symbol an die englischsprachige Bezeichnung anlehnen (aus historischen Gründen wurde diese Regel bisher nicht immer eingehalten).

Nach diesen Regeln sind bis jetzt definiert worden (vergleiche die Informationsausgabe des MAPLIB-Utility-Programms).

Symbol	Eigenschaft	physik.Einheit
VP	Sattdampfdruck	N/M2
VT	Sattdampftemperatur	K
FT	Schmelztemperatur	K

Symbol	Eigenschaft	physik.Einheit
VP	Sattdampfdruck	N/M2
VT	Sattdampftemperatur	K
FT	Schmelztemperatur	K
FH	Schmelzwärme	J/KG
RH	Rekristallisationswärme	J/KG
VH	Verdampfungsenthalpie	J/KG
VS	Verdampfungsentropie	J/KG.K
EH	Enthalpie	J/KG
ES	Entropie	J/KG.K
RO	Dichte	KG/M3
VO	Spezifisches Volumen	M3/KG
PV	Druck als Funktion von Temperaturen und Volumen	N/M2
CP	Spezifische Wärme bei konst. Druck	J/KG.K
CV	Spezifische Wärme bei konst. Volumen	J/KG.K
RS	Spezielle Gaskonstante	J/KG.K
PR	Prandtl.Zahl	1
WL	Wärmeleitfähigkeit	W/M.K
ZD	Zähigkeit, dynamisch	N.S/M2
ZK	Zähigkeit, kinematisch	M2/S
SB	Bruchspannung	N/M2
SD	Dehngrenzspannung	N/M2
SF	Fließgrenzspannung	N/M2
ST	Oberflächenspannung	N/M
BD	Bruchdehnung	1
BE	Brucheinschnürung	1
EM	Elastizitäts Modul	N/M2
GA	Linearer, differentieller Ausdehnungskoeffizient	1/K
RE	Elektrischer Widerstand	OHM
WQ	Wirkungsquerschnitt	M2
CO	Zusammensetzung	1

3. Parameternamen

Grundsätzlich ist mit Ausnahme des verbotenen Namens "I" die Wahl des Parameternamens freigestellt. Der Übersichtlichkeit und Lesbarkeit halber sollten jedoch folgende Regeln berücksichtigt werden

- Parameternamen sollen aus mindestens zwei Zeichen bestehen.
- Für bereits früher verwendete Parameter (siehe Ausgabe des Utility-Programms bei INFO/.PARAMETER) sollten keine neuen Namen definiert werden.
- Für DUMMY-Parameter sollte der Name DUMMY verwendet werden.

4. Systemroutinen- und Common-Namen

Da die Namen der Systemroutinen und der Common Bereiche in keiner Weise verändert werden können, ist für diese das Dollar-Zeichen reserviert.
Subroutinen beginnen mit einem Dollar-Zeichen. Funktionen enden mit einem Dollar-Zeichen.
Common-Namen beginnen und enden mit einem Dollar-Zeichen.

Beispiel:

Subroutine	$NUMBR
Function	NUMBR$
Common	$NUMB$

A n h a n g 2

Die Datenfunktionen sollen folgende Kommentarkarten enthalten: Die Identifikation der Information erfolgt mit mindestens den unterstrichenen Buchstaben.

1. Allgemeine Kommentare, die von MAPLIB-registriert werden.

CNAME	Name des Programms
CAUTOR	Name des oder der Autoren
CDATUM	Datum der Erstellung bzw.der letzten Überarbeitung des Programms
CLITERATUR	zugrundeliegende Literatur
CPARAMETER	Erläuterung der Parameterliste bei Unterprogrammen

2. Spezielle Kommentare für Informationen an MAPLIB

C$MATERIAL	Erläuterung des Materials(Stoffes)
C$PROPERTY	Erläuterung der Eigenschaft
C$REPLACE	Die Routine soll eine vorher integrierte Routine gleichen Namens ersetzen.
C$FIRST-EXECUTABLE-STATEMENT	Diese Karte trennt Deklarationen und ausführbare Anweisungen des Programms.
C$ZAHL	In der Funktion wird noch nicht überprüft, ob die richtige Zahl der Argumente angeliefert wird.

C$GUELTIGKEIT — In der Funktion wird noch nicht überprüft, ob die angelieferten Parameterwerte in ihrem Gültigkeitsbereich liegen.

C$ØMIT — Die folgende Karte soll von MAPLIB nicht auf ihre Bedeutung untersucht werden.

C$SYNONYM — Diese Karte definiert ein Symbol für den Funktionsnamen. In den Masterfunktionen und Registern wird die Funktion nicht mit ihrem eigentlichen Namen sondern dem Synonym geführt.

C$NUMBER — Zahl der Parameter, die mindestens erwartet werden; für die weiteren Parameter sind Standardwerte vorgesehen.

C$TEST-ZONE — Die Karten zwischen C$F und C$T sind Testanweisungen; sie dienen zur Durchführung der Fehlerkontrollen.

C$BYPASS — In der Funktion sind noch keine Anweisungen eingebaut, die das Überspringen der Testzone erlauben.

CR, CZ, C$G und C$B dienen als Anweisungen an das MAPLIB-Utility-Programm, fehlende Statements einzubauen; hierbei werden diese Anweisungskarten aus der Funktion eliminiert.

3. Weitere allgemeine Kommentare, die noch nicht von MAPLIB registriert werden, die aber - soweit sinnvoll - angegeben sein sollten.

CBESCHREIBUNG — Beschreibung des Zwecks und der Anwendungsmöglichkeit des Programms bzw. Verweis auf spezielle Programmbeschreibungen.

CINPUT — Eingabebeschreibung.(Bei Datenfunktionen nicht sinnvoll).

C$\underline{\text{Ø}}$UTPUT Ausgabebeschreibung. (Bei Datenfunktionen nicht sinnvoll).

C$\underline{\text{F}}$ILES Spezifikation der Files.

C$\underline{\text{S}}$UBROUTINES Liste der Unterprogramme, die von dieser Routine benötigt werden, die aber nicht im Paket enthalten sind (wird sehr empfohlen!)

C$\underline{\text{E}}$RROR Erläuterung evtl.Fehlermeldungen (hier meist überflüssig, da Fehlermeldungen selbst erklärend sein sollen).

$\underline{\text{C}}$ Beliebig viele sonstige Kommentarkarten an beliebiger Stelle im Programm.

4. Codierungsregeln für MAPLIB-Kommentarkarten

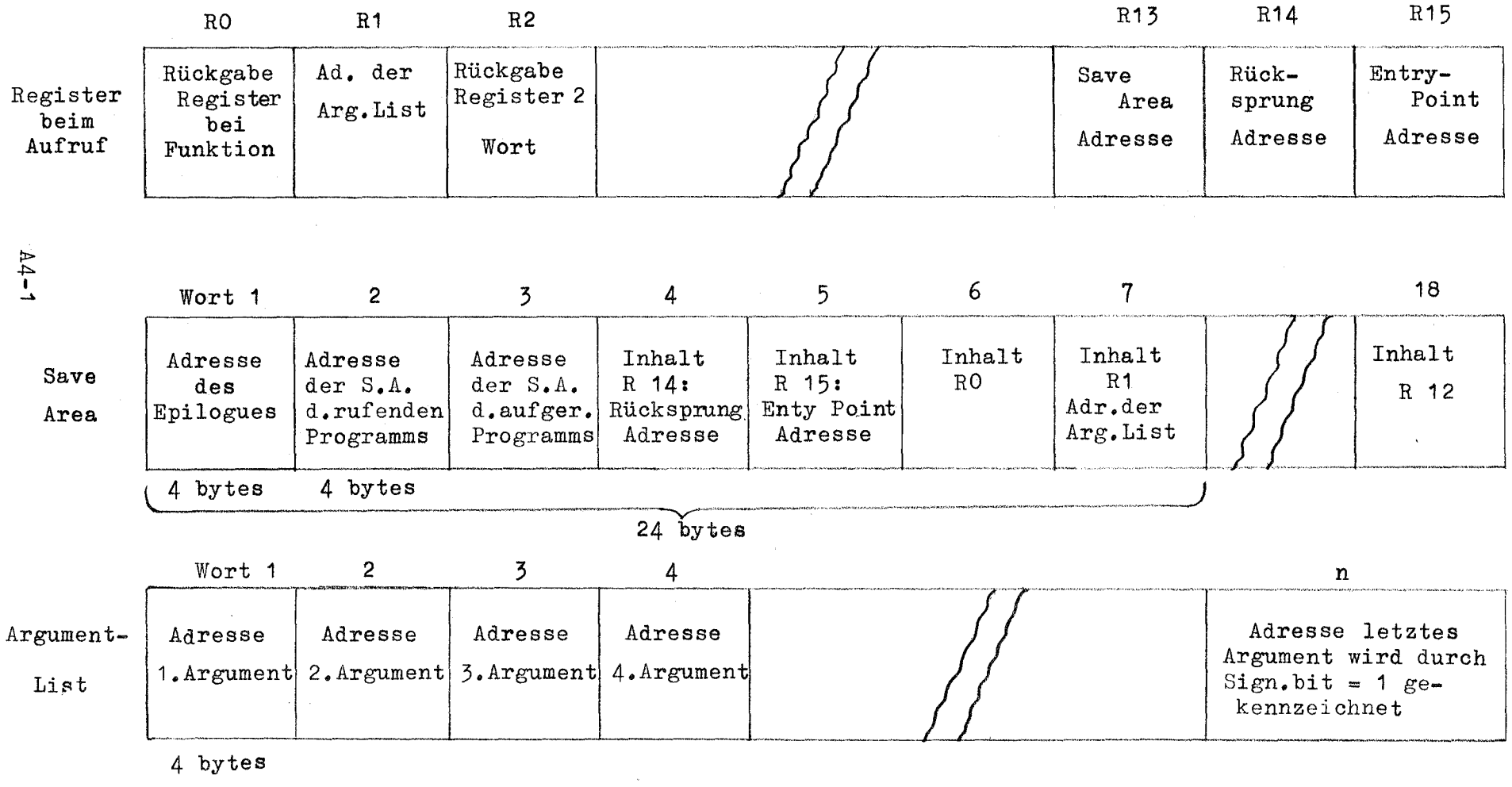
Alle Kommentarkarten beginnen gemäß Fortrankonvention mit dem Buchstaben C in Spalte 1. Die unterstrichene Identifikation ist linksbündig ohne eingeschlossene Blanks zu lochen. Die Detail-Information ist von Spalt 7 bis 72 zu lochen.
Die Identifikation muß auch auf Folgekarten angegeben werden.

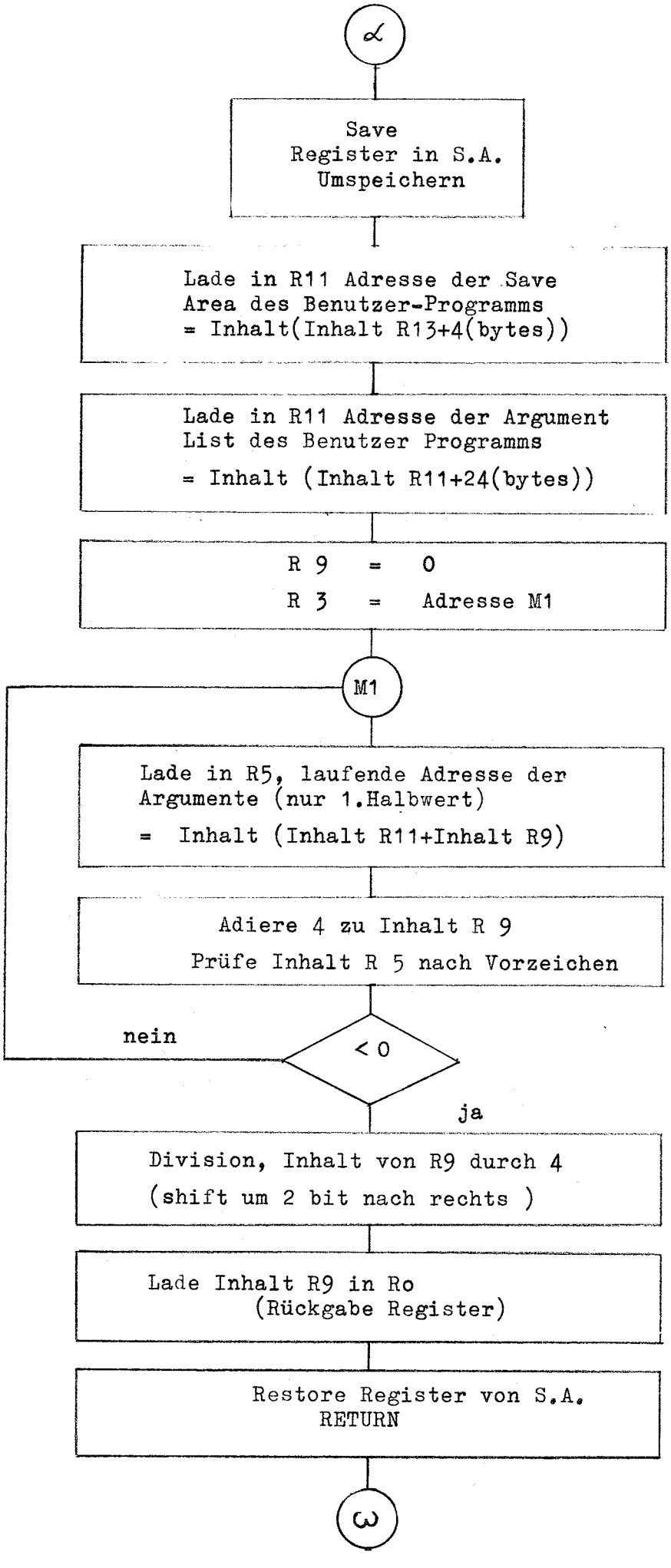
Anhang 3: IBM 360/65-OS-Job Control Language für MAPLIB-Utility

```
//IRE673MA JOB (0673,330,P0000),SCHUMANN,MSGLEVEL=(1,1),CLASS=A,
// MSGCLASS=D
*** JOB-KARTE
//JOBLIB DD DSN=MAPLIB.IRE.UTILITY,UNIT=2314,VOL=SER=GFK006,DISP=SHR
*** DIESER DATA-SET ENTHAELT DAS UTILITY PROGRAMM MIT DEM NAMEN 'MAPLIB'
***
***
***
***
*** MAPLIB-UTILITY-EXECUTION
***
***
//UTILITY EXEC PGM=MAPLIB,REGION=220K,TIME=(2,40)
*** DER TIME PARAMETER IST DEN BEDUERFNISSEN ANZUPASSEN
***
***
//FT01F001 DD DUMMY
*** DUMMY DATA SET ZUR UNTERDRUECKUNG VON AUSGABEN
*** MIT DER UNIT-ANWEISUNG
***
***
//FT06F001 DD SYSOUT=D,DCB=(BLKSIZE=2261,LRECL=133,RECFM=FBA),
// SPACE=(TRK,(100,100))
*** PRINT - DRUCKAUSGABE; WIRD IMMER BENOETIGT
*** DER SPACE PARAMETER IST DEN BEDUERFNISSEN ANZUPASSEN
***
***
//FT07F001 DD SYSOUT=B,SPACE=(3200,(50,20)),
// DCB=(BLKSIZE=3200,LRECL=80,RECFM=FB)
*** PUNCH - KARTEN-AUSGABE; WIRD NUR BEI SOURCE UND LINK BENOETIGT
*** DIE SPACE-ANGABE REICHT FUER 2800 KARTEN
***
***
//FT09F001 DD UNIT=SYSDA,SPACE=(4000,200)
*** DIRECT - DIRECT-ACCESS-EINHEIT; WIRD IMMER BENOETIGT
***
***
//FT11F001 DD UNIT=2314,VOL=SER=GFK006,DISP=(OLD,KEEP),
// DSN=MAPLIB.IRE.SOURCE,DCB=(BLKSIZE=7280,LRECL=80,RECFM=FB)
*** SOURCE - DATEI, DIE SAEMTLICHE FORTRAN ROUTINEN ENTHAELT
*** WIRD BENOETIGT BEI SOUR,LINK,INFO.LIST UND STORE
***
***
//FT13F001 DD UNIT=2314,VOL=SER=GFK006,DISP=(OLD,KEEP),
// DSN=MAPLIB.IRE.REGISTER,DCB=(BLKSIZE=2240,LRECL=80,RECFM=FB)
*** REGISTER        WIRD BENOETIGT BEI OLD UND STORE
***
***
***
*** DIE FOLGENDE KARTE SPEZIFIZIERT DIE INPUT-EINHEIT
*** DIESE KARTE WIRD IMMER BENOETIGT
//FT05F001 DD *
//
```

Flußdiagramm und Liste der Assemblerroutinen NUMBR$

Zum Verständnis des folgenden Flußdiagramms werden hier die Register, die Save Area und die Argumentenliste schematisch dargestellt.




α
Save
Register in S.A.
Umspeichern
Lade in R11 Adresse der Save
Area des Benutzer-Programms
= Inhalt(Inhalt R13+4(bytes))
Lade in R11 Adresse der Argument
List des Benutzer Programms
= Inhalt (Inhalt R11+24(bytes))
R 9 = 0
R 3 = Adresse M1
M1
Lade in R5, laufende Adresse der
Argumente (nur 1.Halbwert)
= Inhalt (Inhalt R11+Inhalt R9)
Adiere 4 zu Inhalt R 9
Prüfe Inhalt R 5 nach Vorzeichen
nein
< 0
ja
Division, Inhalt von R9 durch 4
(shift um 2 bit nach rechts)
Lade Inhalt R9 in Ro
(Rückgabe Register)
Restore Register von S.A.
RETURN
ω

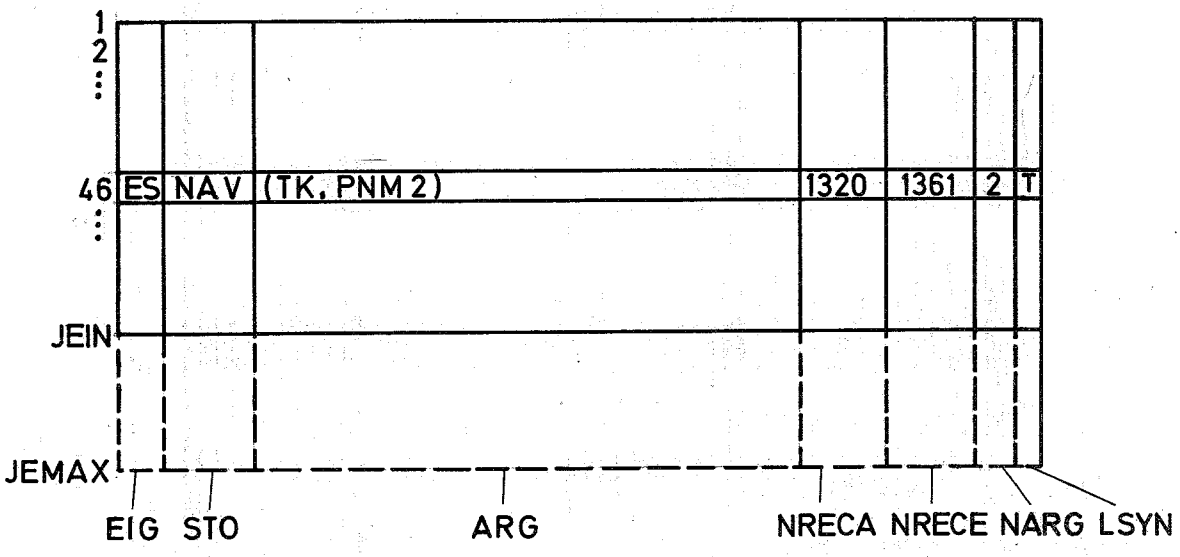
```
NUMER$   START 0
         BC    15,12(15)
         DC    X'7'
         DC    CL7'NUMBR$'
         STM   14,12,12(13)  RETTE REGISTER IN SAVE AREA
         BALR  10,0          SETZEN BASIS REGISTER
         USING *,10          DEFINITION BASIS REGISTER
         L     11,4(13)      LADE R11 ADDR SAVE AREA BENUT.PROGM
         L     11,24(11)     LADE R11 ADDR ARG.LIST BENUT.PROGM
         SR    9,9           SETZE R9 = 0
         LA    3,*+4         ADDR NAECHSTEN BEFEHL IN R3
         LH    5,0(11,9)     LADE R5 ADDR ARGUMENT BENUTZ. PROGM
         LA    9,4(9)        ADDIERE 4 IN R9 : NAECHSTEN ARGUMENT
         LTR   5,5           PRUEFE R5
         BCR   2,3           FALLS > 0 GO TO ADDR IN R3 : LOOP
         SRA   9,2           DIVISION DURCH 4  IN R9
         LR    0,9           LADE INHALT R9 IN R0
         LM    1,12,24(13)   RESTORE REGISTER VON SAVE AREA
         MVI   12(13),X'FF'  SETZE RETURN CODE
         BCR   15,14         RETURN
         END
```

A n h a n g 5

C O M M O N - Register im M A P L I B - Utility - Programm

Für einen schnellen Zugriff zu den auf der SOURCE-Einheit befindlichen MAPLIB-Routinen sind wichtige Teilinformationen über die Routinen in den COMMON-Bereichen des Utility-Programms abgespeichert.
Hier seien die verschiedenen COMMON-Felder erläutert. Die Felder sind im folgenden schematisch dargestellt. Die Breite der gezeichneten Felder entspricht dem reservierten Speicherplatz. Als Beispiel sind die Teilinformationen für die Datenfunktion ESNAV und die Systemroutine ØWARN eingetragen. Um auch für Synonyme ein Beispiel zeigen zu können, wurde angenommen, daß die Datenfunktion ESNAV mit dem Synonym ABCDEF in MAPLIB integriert wurde. Variablen, die Recordnummern enthalten, beziehen sich auf entsprechende Records der Routinen auf der SOURCE Einheit.

1a) Register der Datenfunktionen



JEIN Zahl der integrierten Datenfunktionen

JEMAX Maximal mögliche Zahl der Datenfunktionen

EIG Eigenschaftssymbolteil des Funktionsnamens

STO Stoffsymbolteil des Funktionsnamens

ARG Argumentenliste

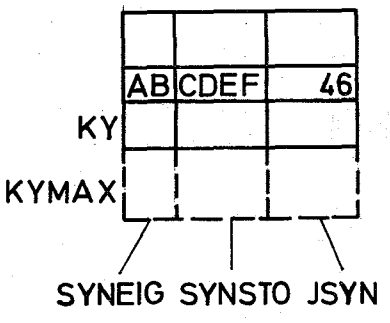
NRECA Anfangs-Record

NRECE End-Record

NARG Zahl der Argumente

LSYN Logische Variable; gibt an, ob es wahr ist, daß die Funktion mit einem Synonym integriert ist.

1b) Register der Synonyme



KY Zahl der registrierten Synonyme

KYMAX Maximal mögliche Zahl der Synonyme

SYNEIG Eigenschaftssymbolteil des Synonymnamens

SYNSTO Stoffsymbolteil des Synonymnamens

JSYN Laufende Nummer der Funktion im Register der Datenfunktionen (siehe oben)

2) Register der Stoff- und Eigenschaftssymbole und der Masterfunktionen

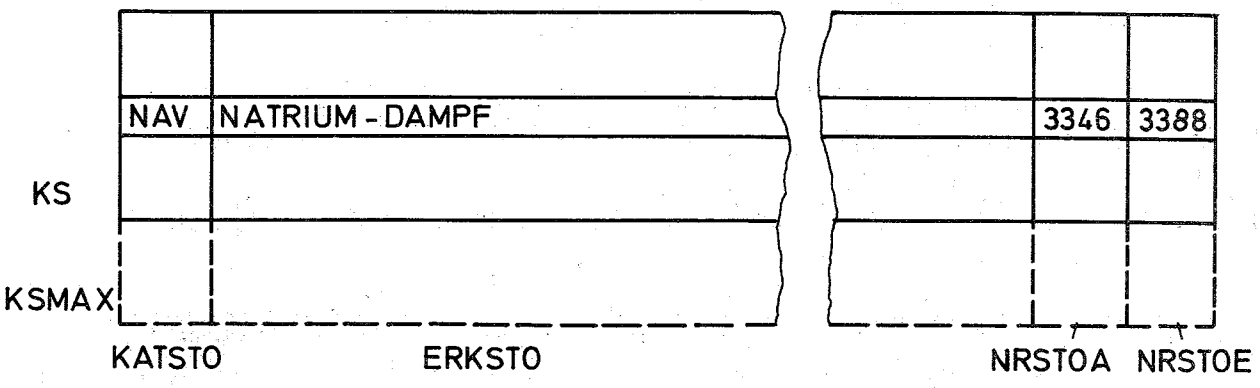
2a) Oberste Masterfunktion

2989	3036
NSSA	NSSE

NØØA Anfangs-Record der obersten Masterfunktion

NØØE End-Record der obersten Masterfunktion

2b) Stoff-Symbole und Masterfunktionen der Stoffebene



KS Zahl der registrierten Stoffe

KSMAX Maximal mögliche Zahl der Stoffe

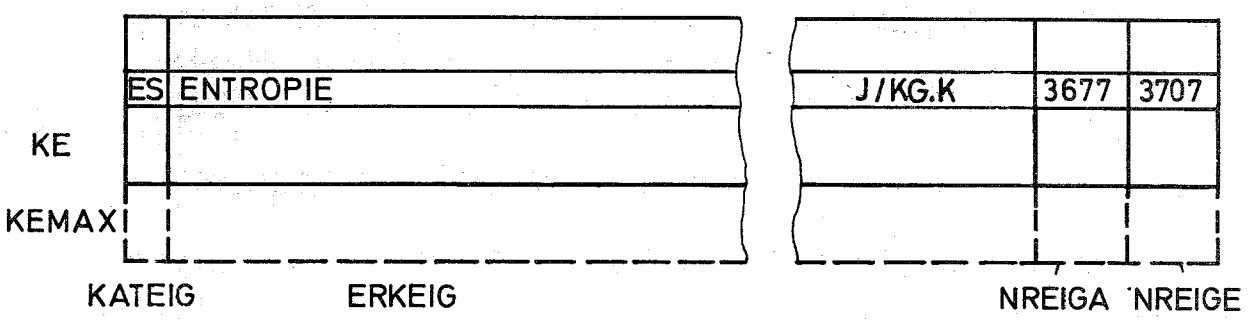
KATSTO Katalog der Stoffsymbole

ERKSTO Erklärung der Stoffsymbole

NRSTOA Anfangs-Record der Masterfunktion der Stoffebene

NRSTOE End-Record

2c) Eigenschafts-Symbole und Masterfunktionen der Eigenschaftsebene



KE Zahl der registrierten Eigenschaften

KEMAX Maximal mögliche Zahl der Eigenschaften

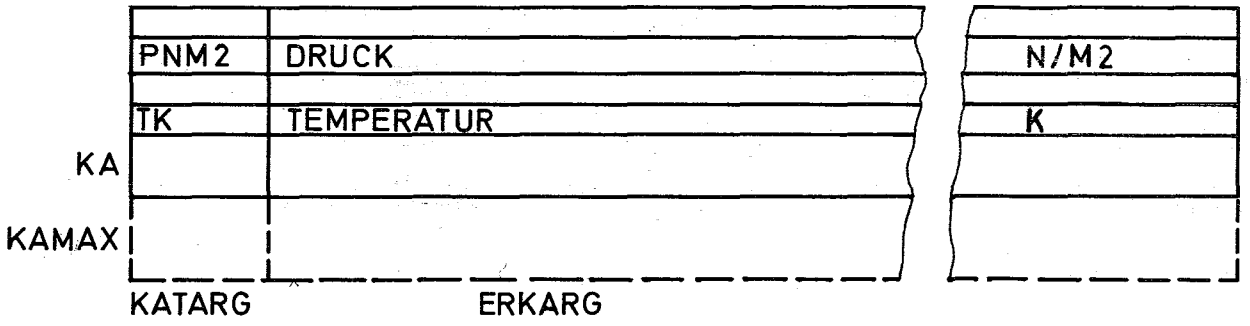
KATEIG Katalog der Eigenschaftssymbole

ERKEIG Erklärung der Eigenschaftssymbole mit Angabe der physikalischen Einheit

NREIGA Anfangs-Record der Masterfunktion der Eigenschaftsebene

NREIGE End-Record

3) Parametersymbole



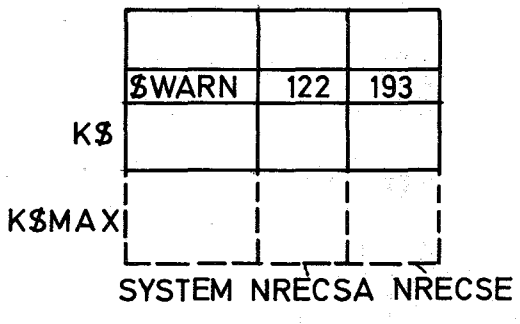
KA Zahl der registrierten Parametersymbole

KAMAX Maximal mögliche Zahl der Parametersymbole

KATARG Katalog der Parametersymbole

ERKARG Erklärung der Parametersymbole mit Angabe der physikalischen Einheit

4) Register der Systemroutinen



K$ Zahl der registrierten Systemroutinen

K$MAX Maximal mögliche Zahl der Systemroutinen

SYSTEM Name der Systemroutine

NRECSA Anfangs-Record der Systemroutine

NRECSE End-Record der Systemroutine

A N H A N G 6

LISTE DER MAPLIB-SYSTEM-ROUTINEN

```
      SUBROUTINE $WARN  (TEXT,VALUES,I,KK)
CN    $WARN
CN    MIT ENTRIES $CATAL,$MESS,$STOP
CB    HIER WERDEN FEHLER REGISTRIERT, NACHRICHTEN AUSGESCHRIEBEN UND
CB    BEI AUFRUF VON $STOP ODER BEI UEBERSCHREITUNG DER ZULAESSIGEN
CB    FEHLERZAHL DER JOB ABGEBROCHEN.
CD    15.06.70
CP    TEXT       FORMAT DER FEHLERNACHRICHT
CP    VALUES     AUSZUDRUCKENDE WERTE
CP    I          ANZAHL DER AUSZUDRUCKENDEN WERTE
CP    KK         FEHLERBEREICH; BEI $CATAL IST KK=5
CV    NT1        AUSGABE-EINHEIT
CV    NFALBE     ZULAESSIGE FEHLERZAHL IN DEN BEREICHEN 1-4 INSGESAMT.
CV    NFALSE     TATSAECHLICHE FEHLERZAHL IN DEN BEREICHEN 1-4 INSGESAMT.
CV    NFB(J)     ZAHL DER FEHLER DIE IM BEREICH J AUFGETRETEN SIND
CV    NF(J)       ZAHL DER FEHLER IM BEREICH J, BIS ZU DEM KEINE
CV               FEHLERNACHRICHTEN AUSGEGEBEN WERDEN
CV    NEND(J)     VOM NEND+1-TEN FEHLER IM BEREICH J WERDEN KEINE
CV                NACHRICHTEN MEHR AUSGEGEBEN
CV    NSTP(J)     BEIM NSTP+1-TEN FEHLER IN BEREICH J WIRD DER JOB
CV                ABGEBROCHEN
CV    K           = .FALSE., WENN NFALBE > 10000 GESETZT WURDE
C     WENN K=.FALSE., WIRD DER STATUSINDIKATOR NICHT .FALSE. GESETZT
CV    KIRRE       = .TRUE. AUSSER, WENN AUCH IM BEREICH 5 FEHLER
CV                ZUGELASSEN WURDEN
CV    KSTAT       STATUSINDIKATOR, WIRD .FALSE. GESETZT, WENN EIN FEHLER
CV                ENTDECKT WURDE; WIRD .TRUE. GESETZT AM ANFANG SOWIE
CV                NACH JEDER ABFRAGE IN STAT$ ODER $STAT7/2/1
      INTEGER*2 NFALBE,NFALSE,NF(5),NFB(5),NEND(5),NSTP(5)
      LOGICAL*1 K,KIRRE,KSTAT,FALSE,TRUE
      COMMON/$KONT$/ NT1,NFALBE,NFALSE,NF,NFB,NEND,NSTP,K,KIRRE,KSTAT,
     1FALSE,TRUE
      COMMON/$FORM$/ KFMT1( 4),KFMT2(22),KFMT3( 4)
      INTEGER TEXT
      DIMENSION TEXT(1),VALUES(1)
      INTEGER WARNIG(5)/'(T1,''     WARNING.  '/,MESSAG(5)/'(T1,''     M
     1ESSAGE.  '/
      INTEGER*2 MES/0/,FIRST/1/
      IF(.NOT.K) GOTO 3
      KSTAT=FALSE
      NFALSE=NFALSE+1
    3 NFB(KK)=NFB(KK)+1
      DO 1 N=1,5
    1 TEXT(N)= WARNIG(N)
      GOTO 44
C
      ENTRY $MESS(TEXT,VALUES,I)
      DO 2 N=1,5
```

```
     2 TEXT(N)= MESSAG(N)
       MES=1
       GOTO 44
C
       ENTRY $CATAL(TEXT,VALUES,I)
       KK=5
       NFB(5)=NFB(5)+1
    44 IF(MES.EQ.1) GOTO 36
       IF(NF(KK).GT.10000) GOTO 15
       IF(NFB(KK).LE.NF(KK).OR.NFB(KK).GT.NEND(KK)) GOTO 15
    36 IF(FIRST.EQ.1) WRITE (NT1,50)
    50 FORMAT(1H1,T45,'MAPLIB  ERROR  MESSAGES')
       WRITE(NT1,TEXT) (VALUES(N),N=1,I)
       FIRST=0
       MES=0
    15 IF(.NOT.KIRRE) GOTO 6
       IF(NFB(5).GE.NSTP(5)) GOTO 5
     6 IF(.NOT.K) GOTO 7
       IF(NFALSE.GT.NFALBE) GOTO 5
     7 IF(KK.GE.5) GOTO 8
       IF(NFB(KK).GE.NSTP(KK)) GOTO 5
     8 RETURN
C
       ENTRY $STOP
     5 WRITE (NT1,KFMT1)
       CALL $NEWS(NT1)
       WRITE (NT1,KFMT2) (KK,NFB(KK),KK=1,5)
       WRITE (NT1,KFMT3)
       STOP
C$CMIT
       END
```

```
      BLOCK DATA
C     INITIALISIERUNG DER COMMON-BEREICHE
CB    FEHLER-REGISTER UND STEUERPARAMETER; VERGL. $WARN
      INTEGER*2 NFALBE,NFALSE,NF(5),NFB(5),NEND(5),NSTP(5)
      LOGICAL*1 K,KIRRE,KSTAT,FALSE,TRUE
      COMMON/$KONT$/ NT1,NFALBE,NFALSE,NF,NFB,NEND,NSTP,K,KIRRE,KSTAT,
     1FALSE,TRUE
      DATA NT1/6/
      DATA NFALBE,NFALSE,NF,NFB,NEND,NSTP/1,0,0,0,0,0,0,0,0,0,0,0,
     110,10,10,10,10,0,0,0,0,0/
      DATA K,KIRRE,KSTAT,FALSE,TRUE/.TRUE.,.TRUE.,.TRUE.,.FALSE.,.TRUE./
C
CB    FORMAT DER FEHLERSTATISTIK DIE IN $STOP AUSGESCHRIEBEN WIRD
      COMMON/$FORM$/ KFMT1( 4),KFMT2(22),KFMT3( 4)
      DATA KFMT1/'(''1'',132(''$''))'/
      DATA KFMT2/'(//1X,T49,''SUMMARY OF ERRORS FOR THIS JOB''/''0'
     1',5(2X,''ERROR RANGE NR.'',I1,I6,2X))'/
      DATA KFMT3/'(''0'',132(''$''))'/
C
CB    PARAMETER ZUR KONTROLLE DER ZAHL DER ARGUMENTE; VERGL. $NUMBR
      COMMON/$NUMB$/J,LJ
      LOGICAL*1LJ
      DATA LJ/.TRUE./
C
C     TEST-STEUERPARAMETER-COMMON
      COMMON/$TEST$/ NOTEST
      LOGICAL*1 NOTEST /.FALSE./
      END
```

```
      SUBROUTINE $NEWS (NT1)
CB    SUBROUTINE ZUR AUSGABE VON MAPLIB-NACHRICHTEN
CD    15.06.70
CP    NT1
CP    AUSGABE-EINHEIT (STANDARD: 6)
      REAL*8 DAT/'15.07.70'/
      WRITE(NT1,1000)
 1000 FORMAT('0',T58,'MAPLIB MESSAGES')
      WRITE(NT1,1) DAT
    1 FORMAT('0',T56,'RELEASE DATE: ',A8)
      RETURN
      END
```

```
      SUBROUTINE $ERROR(K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,
     1K15,K16,K17,K18)
CN    $ERROR
CB    MIT DIESER ROUTINE KOENNEN ALLE STEUER-PARAMETER AUF EINMAL
CB    MODIFIZIERT WERDEN; VERGL. $CNT, $TEST.
CP    K1    -> NT1, AUSGABEEINHEIT DER FEHLERMELDUNGEN
CP    K2    -> NFALBE, ZULAESSIGE FEHLERZAHL IN DEN BEREICHEN 1 - 4
CP    INSGESAMT
CP    K3-7  -> NF, ZAHL DER FEHLER, BIS ZU DER KEINE NACHRICHTEN
CP    GESCHRIEBEN WERDEN, IN DEN VERSCHIEDENEN BEREICHEN
CP    K8-12 -> NEND, ZAHL DER FEHLER, NACH DENEN KEINE NACHRICHTEN
CP    MEHR AUSGESCHRIEBEN WERDEN
CP    K13-17-> NSTP, ZAHL DER ZULAESSIGEN FEHLER IN DEN VERSCHIEDENEN
      COMMON /$TEST$/ NOTEST
      LOGICAL K18,NOTEST*1
CP    K18    -> .NOT.NOTEST, WENN K18=.FALSE. IST, WERDEN DIE TEST-
CP    ANWEISUNGEN IN DEN FUNCTIONEN UEBERSPRUNGEN; BEI K18=.TRUE.
CP    WERDEN DIE FEHLERKONTROLLEN DURCHGEFUEHRT.
      NOTEST=.NOT.K18
CP    BEREICHEN
      INTEGER*2 NFALBE,NFALSE,NF(5),NFB(5),NEND(5),NSTP(5)
      LOGICAL*1 K,KIRRE,KSTAT,FALSE,TRUE
      COMMON/$KONT$/ NT1,NFALBE,NFALSE,NF,NFB,NEND,NSTP,K,KIRRE,KSTAT,
     1FALSE,TRUE
      NT1=K1
      IF(K2.GT.10000)K=FALSE
C      IF(NFALBE > 10 000) -> DER STATUSINDIKATOR WIRD NICHT .FALSE.
C      GESETZT
      NFALBE=K2
      NF(1)=K3
      NF(2) = K4
      NF(3) = K5
      NF(4) = K6
      NF(5) = K7
      NEND(1)=K8
      NEND(2)=K9
      NEND(3)=K10
      NEND(4)=K11
      NEND(5)=K12
      NSTP(1)=K13
      NSTP(2)=K14
      NSTP(3)=K15
      NSTP(4)=K16
      NSTP(5)=K17
      IF(K17.NE.0) KIRRE=FALSE
      DO 27 N=1,5
      IF(NSTP(N).LT.NEND(N)) NSTP(N)=NEND(N)
C     0 <= NEND <=NSTP
   27 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE $CNT   (KBER,K1,K2,K3)
CN    $CNT  -   MODIFIKATION DER KONTROLLPARAMETER
CN    MIT ENTRIES $FILE, $NCPRT, $END, $INDEX, $NSTOP
CB    MIT DIESER ROUTINE KANN DER BENUTZER STEUER-PARAMETER NACH
CB    BELIEBEN MODIFIZIEREN; BEI JEDEM AUFRUF JEDOCH NUR EINIGE.
CB    VERGL $ERRCR
CV    BEZUEGLICH DER COMMON-VARIABLEN VERGL. $WARN
      INTEGER*2 NFALBE,NFALSE,NF(5),NFB(5),NEND(5),NSTP(5)
      LOGICAL*1 K,KIRRE,KSTAT,FALSE,TRUE
      COMMON/$KONT$/ NT1,NFALBE,NFALSE,NF,NFB,NEND,NSTP,K,KIRRE,KSTAT,
     1FALSE,TRUE
      NF(KBER)=K1
      NEND(KBER)=K2
      NSTP(KBER)=K3
    1 RETURN
C
      ENTRY $FILE(K0)
      NT1=K0
      GOTO 1
C
      ENTRY $NCPRT(KBER,KANZ)
      NF(KBER)=KANZ
    5 IF(NSTP(KBER).LE.NEND(KBER)) NSTP(KBER)=NEND(KBER)
      GOTO 1
C
      ENTRY $END (KBER,KANZ)
      NEND(KBER)=KANZ
      GOTO 5
C
      ENTRY $INDEX(K0,K1)
      IF(K0.EQ.0) GOTO 10
      IF(K0.GT.10000) K=FALSE
      NFALBE=K0
   10 IF(K1.EQ.0) GOTO 1
      KIRRE=FALSE
      NSTP(5)=K1
      GOTO 1
C
      ENTRY $NSTOP(KBER,KANZ)
      NSTP(KBER)=KANZ
      GOTO 1
      END
```

```
      SUBROUTINE $TEST (TEST)
CA    SCHUMANN
CD    01.07.70
CN    $TEST
CB    MIT DIESER ROUTINE KANN DER TEST-STEUERPARAMETER MODIFIZIERT
CB    WERDEN.
      LOGICAL TEST,NOTEST*1
      COMMON/$TEST$/ NOTEST
      NOTEST=.NOT.TEST
      RETURN
      END
```

```
      FUNCTION TEST$(/DUMMY/)
CN    TEST$
CD    01.07.70
CB    DIESE FUNCTION GIBT AUSKUNFT UEBER DEN ZUSTAND DES
CB    TEST-STEUERPARAMETERS NOTEST
      COMMON/$TEST$/ NOTEST
      LOGICAL TEST$,NOTEST*1
      TEST$=.NOT.NOTEST
      RETURN
      END
```

```
      SUBROUTINE $STAT7(NK,NFK,NF1,NF2,NF3,NF4,NF5)
CN    $STAT7   ABFRAGE DER INDIKATORPARAMETER
CN       MIT ENTRIES $STAT2 UND $STAT1
CD    01.06.70
CP    NK       ERHAELT DEN WERT DES STATUSINDIKATORS
CP    NFK      NUMMER DES HOECHSTEN FEHLERBEREICHES
CP    NF1-5    ANZAHL DER FEHLER IN DEN VERSCHIEDENEN BEREICHEN
CP             SEIT DER LETZTEN ABFRAGE BZW. SEIT BEGINN DER RECHNUNG
      INTEGER*2 NFALBE,NFALSE,NF(5),NFB(5),NEND(5),NSTP(5)
      LOGICAL*1 K,KIRRE,KSTAT,FALSE,TRUE
      COMMON/$KONT$/ NT1,NFALBE,NFALSE,NF,NFB,NEND,NSTP,K,KIRRE,KSTAT,
     1FALSE,TRUE
      INTEGER*2 LFB(5)/5*0/
      LOGICAL NK
      NF1=NFB(1)-LFB(1)
      NF2=NFB(2)-LFB(2)
      NF3=NFB(3)-LFB(3)
      NF4=NFB(4)-LFB(4)
      NF5=NFB(5)-LFB(5)
      DO 2 I=1,5
      IF((NFB(I)-LFB(I)).NE.0) NFK=I
    2 CONTINUE
      GOTO 5
C
      ENTRY  $STAT2(NK,NFK)
      NFK=0
      DO 1 I=1,5
      IF(NFB(I).NE.0) NFK=I
    1 CONTINUE
C
    5 DO 6 I=1,5
      LFB(I)=NFB(I)
    6 CONTINUE
C
      ENTRY $STAT1(NK)
      NK=KSTAT
      KSTAT=TRUE
      RETURN
      END
```

```
      FUNCTION STAT$ (N)
CN    STAT$    - ANGABE DES STATUSINDIKATORS
CD    15.06.70
CB    STAT$ = KSTAT
CV    KSTAT, LOGICAL, STATUSINDIKATOR
CV    IST TRUE WENN SEIT BEGINN DER RECHNUNG ODER SEIT DER LETZTEN
CV    ANFRAGE KEIN FEHLER IM BEREICH 1-4 AUFTRAT;
CV    IST FALSE SONST.
      LOGICAL STAT$
      INTEGER*2 NFALBE,NFALSE,NF(5),NFB(5),NEND(5),NSTP(5)
      LOGICAL*1 K,KIRRE,KSTAT,FALSE,TRUE
      COMMON/$KCNT$/ NT1,NFALBE,NFALSE,NF,NFB,NEND,NSTP,K,KIRRE,KSTAT,
     1FALSE,TRUE
      STAT$=KSTAT
      KSTAT=TRUE
      RETURN
      END
```

```
      FUNCTION ERR$$$(/KSTOFF/)
CN    ERR$$$
CN    MIT ENTRIES ERR$X$ UND ERRX$$
CB    DIESE FUNCTION WIRD VON DEN MASTERFUNKTIONEN BEI EINEM KATALOG-
CB    FEHLER ODER BEI FEHLENDEM STOFF-SYMBOL (IN$$$$$$) AUFGERUFEN;
CB    DIE FUNCTION GIBT DIE FEHLERMELDUNG MIT DEM ENTSPRECHENDEN  FORMAT
CB    WEITER UND LIEFERT DEN STANDARDWERT 1. ZURUECK.
CC    01.06.70
      COMMON/$NUMB$/J,LJ
      LOGICAL*1 LJ
      INTEGER TEXT1(25),TEXT2(18),TEXT3(26),TEXT4(26),V(2)
      INTEGER*2 VALUE(4),V2(4)
      EQUIVALENCE(V2(1),V(1))
      DATA TEXT1/'                          FUNCTION $$$$$$ IS CALLED WITH'',
     1I2,''ARGUMENTS; THERE SHOULD BE AT LEAST TWO'')'/
      DATA TEXT2/'(T1,'' *** WARNING. MATERIAL ''A4,'' IS NOT CATALOGUED
     1, STANDARD VALUE=1.'')'/
      DATA TEXT3            /'(T1,'' *** WARNING.  FUNCTION $$''A4,'' FOR
     1 THE PROPERTY ''A4,''IS NOT CATALOGUED, STANDARD VALUE = 1.'')'/
      DATA TEXT4/'(T1,'' *** WARNING.  FUNCTION ''A2,''$$$$ FOR THE MATE
     1RIAL ''A4,'' IS NOT CATALOGUED, STANDARD VALUE = 1.'')'/
      LJ=.TRUE.
      IF (J.GT.-1) GO TO 1
      J=J+2
      CALL$WARN(TEXT1,J,1,4)
      GO TO 2
    1 CALL$CATAL(TEXT2,KSTOFF,1)
    2 ERR$$$=1.
      RETURN
C
      ENTRY ERR$X$(VALUE)
      DO 3 I=1,4
    3 V2(I)=VALUE(I)
      CALL$CATAL(TEXT3,V,2)
      GO TO 2
C
      ENTRY ERRX$$(VALUE)
      DO 4 I=1,4
    4 V2(I)=VALUE(I)
      CALL$CATAL(TEXT4,V,2)
      GO TO 2
      END
```

```
      SUBROUTINE $NUMBR(Y,KPROG,J1,JSOLL,M,I,*)
CN    $NUMBER - KONTROLLE DER ARGUMENTEN-ANZAHL
CD    01.06.70
CP    Y       WERT DER FUNCTION; Y=1. WENN ZUWENIG ARGUMENTE DA SIND
CP    KPROG   NAME DER FUNCTION
CP    J1      ANZAHL DER ARGUMENTE, WIE SIE MIT NUMBER$ IN DER FUNCTION
CP            FESTGESTELLT WURDE
CP    JSOLL   VEKTROR DER ZULAESSIGEN ARGUMENTENZAHLEN
CP    M       LAENGE DES VEKTORS JSOLL
CP    I       INDEX MIT JSOLL(I)<= J1
CP    *       RUECKSPRUNGLABEL FUER ZU WENIG ARGUMENTE
C
      COMMON/$NUMB$/ J,LJ
      LOGICAL*1 LJ
CV    LJ      =.FALSE., WENN ZAHL DER ARGUMENTE BEREITS IN MASTER-FUNC-
CV            =.TRUE. ,WENN DIE DATENFUNCTION DIRECT AUFGERUFEN WURDE
CV                    TION FESTGESTELLT WURDE
CV    J       ANZAHL DER ARGUMENTE BEI AUFRUF EINER MASTER-FUNCTION
      INTEGER TEXT1(25)/'                          FUNCTION ''A4,A2,X,''IS CAL
     1LED WITH'',I3,''ARGUMENTS, THESE ARE MORE THAN NEEDED'')'/
      INTEGER TEXT2(29)/'                          FUNCTION ''A4,A2,X,''IS CAL
     1LED WITH'',I2,''ARGUMENTS, THESE ARE NOT ENOUGH, STANDARD VALUE =
     21.'')'/
      DIMENSION KPROG(2),JSOLL(M),KVAL(3)
C
      IF (.NOT. LJ) GOTO 5
C
C   LJ = .TRUE.
C
      I=1
      J=J1
      IF (JSOLL(1) .EQ. 0 .AND. J .EQ. 1) RETURN
C
C   RETURN.. DIE FUNKTION WURDE MIT EINEM DUMMY-ARGUMENT AUFGERUFEN
C
    5 DO 1 I=1,M
      IF (J .EQ. JSOLL(I)) RETURN
C
C   RETURN.. DIE ARGUMENTENZAHL STIMMT MIT DER I-TEN MOEGLICHEN
C            ARGUMENTENZAHL UEBEREIN
    1 CONTINUE
      KVAL(1)=KPROG(1)
      KVAL(2)=KPROG(2)
      KVAL(3)=J
      IF ( J.GT. JSOLL(1)) GO TO 2
C
C   DIE ARGUMENTENZAHL IST KLEINER ALS ERFORDERLICH
C   FEHLERBEREICH 4
      Y=1.
      CALL$WARN(TEXT2,KVAL,3,4)
      RETURN 1
C
    2 IF (M .LE.  1) GO TO 6
      M1=M-1
      DO 3 I=1,M1
      IF (J .GT. JSOLL(I) .AND. J.LT.JSOLL(I+1)) GO TO 4
C
```

```
C   GOTO4..   JSOLL(I) < J < JSOLL(I+1)
C             FEHLERBEREICH 1
    3 CONTINUE
C
C   GOTO6..   JSOLL(M) < J
C             FEHLERBEREICH 1
    6 I=MAXO(M,1)
    4 CALL$WARN(TEXT1,KVAL,3,1)
      RETURN
      END
```

```
      SUBROUTINE $RANGE(PROG,KPROG,NARG,AMIN3,AMIN2,ARG,AMAX2,AMAX3,
     1NDIM,*)
CA    $RANGE
CD    01.05.70
CB    $RANGE TESTS WETHER THE PARAMETER WITH THE NAME NARG AND THE VALUE
CB    ARG IS IN THE PERMISSIBLE RANGE BETWEEN AMIN2 AND AMAX2 OR AT
CB    LEAST BETWEEN AMIN3 AND AMAX3;IN THE LATER CASE IT CAUSES BY $TEXT
CB    2 AN ERROR MESSAGE DUE TO ERROR-RANGE NR.2.
CB    IF ARG IS OUTSIDE THE RANGE AMIN3 TO AMAX3 IT CAUSES BY $TEXT3 AN
CB    ERROR MESSAGE DUE TO ERROR-RANGE NR.3., PROG=1. AND RETURN BY *
CP    PROG  = VALUE OF CALLING FUNCTION
CP    KPROG = NAME  OF CALLING FUNCTION
CP    NARG  = NAME  OF PARAMETER
CP    ARG   = VALUE OF PARAMETER
CP    NDIM  = UNIT  OF THE PARAMETER; E.G.' N/M**2'IF ARG=PRESSURE
CP    AMIN3 =
CP    AMIN2 =
CP    AMAX2 =          AMIN3<=AMIN2<= ARG <= AMAX2 <= AMAX3
CP    AMAX3 =
      DIMENSION KPROG(2),NARG(2),NDIM(3)
      IF(AMIN2.LE.ARG.AND.ARG.LE.AMAX2) RETURN
      IF(ARG.LE.AMAX3.AND.ARG.GE.AMIN3) GOTO 3
      CALL $TEXT3(KPROG,NARG,AMIN3,AMAX3,ARG,NDIM)
      PROG=1.
      RETURN 1
    3 CALL $TEXT2(KPROG,NARG,AMIN2,AMAX2,ARG,NDIM)
      RETURN
      END
```

```
      SUBROUTINE $WARN2 (KPROG,IARG,AMIN,AMAX,ARG)
CB    THESE ROUTINES ARE CALLED BY $RANGE AND RANGE AND RAGVL1;
CB    REFER THESE ROUTINES FOR EXPLANATION
      DIMENSION KPROG(2),NTEXT(27),KVAL(16),VAL(16),NFALL(12)
      DIMENSION NARG(2),NDIM(3)
      INTEGER AX/ 5/
C     AX = MAXIMUM NUMBER OF CATALOGIZED ARGUMENTS
      INTEGER KARG(2, 5)/'TK      PNM2    ZSEC    CONCTR  PORVOL  '/
      INTEGER KDIM(3, 5)/'GRAD K      N/M**2      SEC          MOL-PROCEN
     1T VOL.-PROCENT'/
      EQUIVALENCE (KVAL(1),VAL(1))
      DATA NFALL/',VALUE MAY BE UNRELIABLE,STANDARD VALUE = 1.    '/
      DATA NTEXT/'                        ARGUMENT '',A4,A2,''='',G11.4,'' O
     1UT OF RANGE'',G11.4,'' TO'',G11.4,1X,3A4,'' IN '',A4,A2,6A4)'/
      DO 3 I= 11,16
    3 KVAL(I)=NFALL(I-10)
      I=2
      GO TO 2
C
      ENTRY $WARN3 (KPROG,IARG,AMIN,AMAX,ARG)
      DO 1 I= 11,16
    1 KVAL(I)=NFALL(I-4)
      I=3
    2 IF(IARG.LT.1.OR.IARG.GT.AX) GOTO 998
      KVAL(1)=KARG(1,IARG)
      KVAL(2)=KARG(2,IARG)
      KVAL(6)= KDIM(1,IARG)
      KVAL(7)= KDIM(2,IARG)
      KVAL(8)=KDIM(3,IARG)
   14 VAL(3)=ARG
      VAL(4) = AMIN
      VAL(5) = AMAX
      KVAL(9)= KPROG(1)
      KVAL(10)=KPROG(2)
      CALL $WARN (NTEXT,KVAL,16,I)
  999 RETURN
  998 WRITE(6,100) IARG,KPROG
  100 FORMAT('0THE ARGUMENT WITH NR.',I3,' CALLED IN ',2A4,'IS NOT CATAL
     1OGIZED')
      GO TO 999
C
      ENTRY $TEXT3(KPROG,NARG,AMIN,AMAX,ARG,NDIM)
      DO 11 I=11,16
   11 KVAL(I)=NFALL(I-4)
      I=3
      GOTO 12
C
      ENTRY $TEXT2(KPROG,NARG,AMIN,AMAX,ARG,NDIM)
      DO 13 I=11,16
   13 KVAL(I)=NFALL(I-10)
      I=2
   12 KVAL(1)=NARG(1)
      KVAL(2)=NARG(2)
      KVAL(6)=NDIM(1)
      KVAL(7)=NDIM(2)
      KVAL(8)=NDIM(3)
      GOTO 14
```

```
END
```

```
      SUBROUTINE RANGE (PROG,KPROG,IARG,AMIN2,AMAX2,AMIN3,AMAX3,ARG,*)
CN    RANGE   TEST OF ARG
CD    01.05.70
CB     RANGE TESTS WETHER THE PARAMETER WITH THE                    VALUE
CB    ARG IS IN THE PERMISSIBLE RANGE BETWEEN AMIN2 AND AMAX2 OR AT
CB    LEAST BETWEEN AMIN3 AND AMAX3;IN THE LATER CASE IT CAUSES BY $TEXT
CB    2 AN ERROR MESSAGE DUE TO ERROR-RANGE NR.2.
CB    IF ARG IS OUTSIDE THE RANGE AMIN3 TO AMAX3 IT CAUSES BY $TEXT3 AN
CB    ERROR MESSAGE DUE TO ERROR-RANGE NR.3., PROG=1. AND RETURN BY *
CP    PROG   = VALUE OF CALLING FUNCTION
CP    KPROG  = NAME OF CALLING FUNCTION
CP    IARG   = NUMBER OF CATALOGUED NAME OF ARGUMENT; SEE $WARN2
CP    AMIN2  =
CP    AMAX2  =
CP    AMIN3  =  AMIN3<=AMIN2<=ARG<=AMAX2<=AMAX3
CP    AMAX3  =
CP    ARG    VALUE OF ARGUMENT
CP    *      = RETURN LABEL IF ARG IS OUT OF RANGE AMIN3 TO AMAX3
      DIMENSION KPROG(2)
      IF(AMIN2.LE.ARG.AND.ARG.LE.AMAX2) RETURN
      IF(ARG.LE.AMAX3.AND.ARG.GE.AMIN3) GO TO 3
      CALL $WARN3(KPROG,IARG,AMIN3,AMAX3,ARG)
      PROG= 1.
      RETURN 1
    3 CALL $WARN2(KPROG,IARG,AMIN2,AMAX2,ARG)
      RETURN
      END
```

```
      FUNCTION RAGVL1(KPROG,IARG,IFORM,IA,AMIN2,AMAX2,AMIN3,AMAX3,ARG,A)
CN    RAGVL1
CA    SCHUMANN
CD    01.01.70
C
CB    FUNCTION TO TEST THE RANGE OF THE ARGUMENT ARG AND TO COMPUTE THE
CB    VALUE OF THE FUNCTION KPROG
C
CP    KPROG  = NAME OF CALLING FUNCTION
CP    IARG   = NUMBER OF THE CATALOGUED NAME OF ARGUMENT
CP    IFORM  = NUMBER OF COMPUTING-FORMULA
CP    IA     = LENGTH OF VECTOR A
CP    AMIN2  =
CP    AMAX2  = LIMITS OF PERMISSIBLE RANGE OF ARGUMENT ARG; OUT OF THEM
CP    AMIN3  = THERE WILL BE AN ERROR OF TYP NR.2 OR 3
CP    AMAX3  =
CP    ARG    = VALUE OF ARGUMENT
CP    A      = COEFFICIENTS VECTOR
C
      COMMON/$TEST$/ NOTEST
      LOGICAL*1 NOTEST
      DIMENSION KPROG(2),A(1)
      INTEGER FX/1/
C
C     FX = MAXIMUM NUMBER OF CATALOGIZED COMPUTING-FORMULA
C     TEST OF RANGE
C
      IF(NOTEST) GOTO 5
      IF(AMIN2.LE.ARG.AND.ARG.LE.AMAX2) GOTO 4
      IF(ARG.LE.AMAX3.AND.ARG.GE.AMIN3) GOTO 3
      CALL $WARN3(KPROG,IARG,AMIN3,AMAX3,ARG)
C     ERROR-RANGE NR.3
  997 RAGVL1= 1.
C
C     STANDARD-VALUE, DUE TO ERROR-RANGE NR.3
C
      GOTO 999
    3 CALL $WARN2(KPROG,IARG,AMIN2,AMAX2,ARG)
C     ERROR-RANGE NR.2
C
    4 IF(IFORM.LT.0.OR.IFORM.GT.FX) GOTO 998
    5 I=IFORM+1
C
C     COMPUTING THE FUNCTION-VALUE
C
      GO TO (996,21,22,23,24,25),I
   21 CALL PVAL(W,ARG,A,IA)
C     PVAL IS A SUBROUTINE OUT OF THE FORTRAN-LIBRARY;
C     IT COMPUTES THE VALUE W OF AN POLYNOM
C     W = A(1) +A(2)*ARG + A(3)*ARG**2 + ... + A(IA)*ARG**(IA-1)
      RAGVL1 = W
      GOTO 999
   22 CONTINUE
C
C     HERE YOU CAN ADD ANOTHER COMPUTING-FORMULA
C
   23 CONTINUE
```

```
   24 CONTINUE
   25 CONTINUE
  998 WRITE(6,100) KPROG,IFORM
  100 FORMAT('0IN RAGVL1, CALLED BY ',2A4,', THE FORMULA WITH NR.',I3, '
     1DOES NOT EXIST ')
      GO TO 997
  996 RAGVL1=0.
C
C     THIS IS THE EXIT FOR  IFORM=0, IF ARG IS NOT OUT OF RANGE 3
C
  999 RETURN
      END
```

A N H A N G 7

LISTE DES MAPLIB-UTILITY-PROGRAMMS

```
CALL MAPLIB
STOP
END
```

```
      SUBROUTINE MAPLIB
CN    MAPLIB   - STEUERROUTINE
CA    SCHUMANN
CV
CV    COMMON - BEREICHE
CV
CV    NAMENSGEBUNG   SXXXN
CV         S  ZUR KENNZEICHNUNG EINES COMMON-NAMENS
CV         XXX  = DIM   BEI FELDERN
CV              = KONT  BEI SKALAREN
CV         N    = LAENGEN-SPEZIFIKATION  DER VARIABLEN
C
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTOA(100),NRSTOE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
CV
CV    STO      LISTE DER STOFFSYMBOLE IN DER REIHENFOLGE DER FUNCTIONS
CV             VERGL. EIG
CV    ARG      ARGUMENTEN-LISTE DER FUNCTIONS (JEWEILS 24 ZEICHEN)
CV    NRECA    RECORD-NR DES FUNCTION-BEGINNS AUF DEM FILE NF1
CV    NRECE    RECORD-NR DES FUNCTION-ENDES   AUF DEM FILE NF1
CV    KATSTO   KATALOG DER STOFFE
CV    SYNSTO   STOFF-SYNONYME
CV    JSYN     FUNCTION-NR DIE ZU DIESEM SYNONYM GEHOERT
CV    ERKEIG   ERKLAERUNG DER EIGENSCHAFTEN
CV    ERKSTO   ERKLAERUNG DER STOFFE
CV    ERKARG   ERKLAERUNG DER ARGUMENTE
CV    NRSTOA   RECORD-NR DES BEGINNS DER MASTER-FUNCTION DER STOFF-EBENE
CV    NRSTOE   RECORD-NR DES ENDES   DER MASTER-FUNCTION DER STOFF-EBENE
CV    NREIGA     DITO. FUER EIGENSCHAFTS-EBENE
CV    NREIGE     DITO. FUER EIGENSCHAFTS-EBENE
CV    NRECSA     DITO. FUER SYSTEM-ROUTINEN
CV    NRECSE     DITO. FUER SYSTEM-ROUTINE
C
      COMMON/SDIM2/ EIG(500),NARG(500),KATEIG(100),SYNEIG(20),KATARG
     1(3,50),SYSTEM(3,50)
      INTEGER*2 EIG,NARG,KATEIG,SYNEIG,KATARG,SYSTEM
CV    EIG      LISTE DER EIGENSCHAFTSSYMBOLE IN DER REIHENFOLGE DER
CV             FUNCTIONS.  JEWEILS  EIN  ELEMENT AUS EIG UND EINS AUS
CV             STO (SIEHE OBEN) ENTHALTEN GEMEINSAM DEN NAMEN DER FUNC-
CV             TION
CV    NARG     ANZAHL DER ARGUMENTE DER FUNCTIONS
CV    KATEIG   KATALOG DER EIGENSCHAFTEN
CV    SYNEIG   EIGENSCHAFTS-SYNONYM
CV    KATARG   KATALOG DER ARGUMENTEN-SYMBOLE
CV    SYSTEM   KATALOG DER SYSTEM-ROUTINEN-NAMEN
C
      COMMON/SDIM1/ LSYN(500),LERL(4000),FIL(80,100)
      LOGICAL*1LSYN,LERL,FIL
CV    LSYN     IST .TRUE., WENN EINE FUNCTION EINEN SYNONYM-NAMEN ENT
CV             HAELT; NORMAL: .FALSE.
CV    LERL     IST EIN INDIKATOR, DER BEIM SCHREIBEN DER EINGABE AUF
CV             DIRECT-ACCESS .FALSE. GESETZT WIRD; BEI DER ABARBEITUNG
CV             DER KARTEN AUF DIRECT-ACCESS WIRD DER INDIKATOR AUF .TRUE.
CV             GESETZT, WENN DER ENTSPRECHENDE RECORD ABGEARBEITET WURDE
```

```
CV            (VERGL. FILFIL, ADDFUN, FUNANA, KATALO)
CV     FIL    ARBEITSFELD; DIENT ALS BUFFER BEIM LESEN VON DIRECT-ACCESS
CV            IN  GETREC UND BEIM SCHREIBEN  IN FILFIL
C
      COMMON/SKCNT8/RELDAT,RELUHR,DAT,UHR
      REAL*8 RELDAT,RELUHR,DAT,UHR
CV    RELDAT  DATUM DER LETZTEN AENDERUNG DER SOURCE-DATEI
CV    RELUHR  UHRZEIT "     "        "      "     "
CV    DAT     AKTUELLES DATUM;     GEMAESS  CALL DATUM(DAT,UHR)
CV    UHR     AKTUELLE UHRZEIT                  (SIEHE PUBLIC)
C
      COMMON/SKCNT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
CV    NDIMAX  ANZAHL DER KARTEN, DIE IN FIL HINEINPASSEN (100)
CV    NBRREC  ZAHL DER RECORDS AUF DIRECT-ACCESS (SPACE-ANGABE)  (200)
CV    NDA     DIRECT-ACCESS-EINHEIT (9)
CV    NRLEN   KARTEN-LAENGE (80)
CV    JEINO   ZAHL DER INTEGRIERTEN FUNKTIONEN BEI JOB-BEGINN
CV    JEIN    AKTUELLE ZAHL DER INTEGRIERTEN FUNKTIONEN
CV    JEMAX   MAXIMAL ZULAESSIGER WERT VON JEIN  (500)
CV    KE      AKTUELLE ZAHL DER EIGENSCHAFTEN
CV    KEO     ZAHL DER EIGENSCHAFTEN BEI JOB-BEGINN
CV    KEMAX   MAXIMAL ZULAESSIGER WERT VON KE    (100)
CV    KS
CV    KSO         ANALOG FUER STOFFE
CV    KSMAX                                     (100)
CV    KA
CV    KAO         ANALOG FUER ARGUMENTE
CV    KAMAX                                     ( 50)
CV    KY          ANALOG FUER SYNONYME
CV    KYO
CV    KYMAX                                     ( 20)
CV    K$          ANALOG FUER SYSTEM-ROUTINEN
CV    K$O
CV    K$MAX                                     ( 50)
CV    INP     INPUT-EINHEIT   (5)
CV    OUT     PRINT-EINHEIT   (6)
CV    PRT     PUNCH-EINHEIT   (7)
CV    NF1     SOURCE-EINHEIT  (11)
CV    NF2     EINHEIT DER MASTER-FUNCTIONS;  NF2=NF1 (11)
CV    NF3     REGISTER-EINHEIT (13)
CV    NRF11   ZAHL DER RECORDS AUF DER SOURCE-EINHEIT
CV    IAV     ASSOCIATED VARIABLE  (DIRECT ACCESS)
CV    N$$A    RECORD-NR. DES BEGINNS  BZW. DES ENDES DER OBERSTEN
CV    N$$E    MASTER-FUNCTION
C
      COMMON/SKCNT2/ NAMAX,NAMAXO
      INTEGER*2 NAMAX,NAMAXO
CV    NAMAX   MAXIMUM DER ARGUMENTENZAHL (AKTUELL)
CV    NAMAXO  DITO. ZU BEGINN DES JOBS
C
      COMMON/SKCNT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1   ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,  FUNCDA,NEWMAS,PUNCH
```

```
CV       IM AUSGANGS- UND NORMALFALL SIND ALLE DIESE LOGISCHEN VARIABLEN
CV            .FALSE.
CV    ERROR   =.TRUE., WENN EIN SCHWERER FEHLER AUFTRAT
CV    FERROR  =.TRUE., WENN EIN EINGABEFEHLER ENTDECKT WURDE
CV    NURSOR  = LINK  (PER EQUIVALENCE);IST .TRUE. NACH KEY LINK
CV    NOTARG  =.TRUE., WENN AUF DEM SOURCE-FILE LUECKEN BESTEHEN, DIE
CV             MIT DER ROUTINE SCRA ELIMINIERT WERDEN KOENNEN
CV    NOTEIG        OFFEN
CV    NOTSTO  =.TRUE., WENN SPALTE 73-80 BEI AUSGABE NICHT DURCHNUMME
CV             RIERT WERDEN SOLL
CV    NOTDOU  = LNCC (PER EQUIVALENCE); IST .TRUE. NACH .NOC: AUSGABE
CV              OHNE KOMMENTARKARTEN
CV    ERKFOL  =.TRUE.  NACH KEY COMMENT
CV    FUNCDA  =.TRUE., WENN KOPIE DES SOURCE-FILE AUF DIRECT-ACCESS IST
CV    NEWMAS  =.TRUE., WENN NEUERSTELLUNG DER MASTERFUNCTIONS ERFORDERL.
CV    PUNCH   =.TRUE., WENN KARTEN ZU STANZEN SIND; NACH KEY SOURCE UND
CV              NACH LINK
C
C
C
C
CV    LOKALE  VARIABLE
      REAL*8 W8,PASSW
      INTEGER WORT(2),NEW/'NEW '/,OLD/'OLD '/,ENDE/'END '/,STOFF,
     1STEUER(10)/'DELE','COMM','STOR','ADD ','INFO','SOUR','LINK','UNIT'
     2,'NAME','COPY'/  ,
     2STINFO(12)/'.ALL','.LIS','.MAP','.MAS','.MAT','.PRO','.PAR',
     3'.TAB','.FUN','.SYS','.NEW','.NOC'/
     6 ,STLINK(5)/'.JOB','.DSN','.COM','.TIM','.JID'/
       LOGICAL*1 GETFIL          ,OUTPUT,$     ,END/.FALSE./        ,LINK
     1,LNEW,LNCC
      INTEGER*2     EIGEN(4)
     1   ,      $2/'$ '/,DOLLAR/'$ '/
      EQUIVALENCE (STOFF,WORT(2)),(EIGEN(1),WORT(1)),(W8,WORT(1)),
     2 (NURSOR,LINK),(LNCC,NOTDOU)
     3          , ($,$2)
      DATA IKEYER/0/
C
CV    IKEYER  ZAEHLER FUER SCHLUESSELWORT-FEHLER
C
C
      CALL FSPIE
CS    FSPIE  - EIN PROGRAMM ZUR BEHANDLUNG VON PROGRAMM-UNTERBRECHUNGEN
CS    BEI FORTRAN-PROGRAMMEN; J.ENZMANN; GESELLSCHAFT FUER KERNFORSCHG.
CS    DVZ-ARBEITSBERICHT NR.31, PROGRAMMBESCHREIBUNG NR.209
C
      DEFINE FILE 9 (200,1000,U,IAV)
C     DIRECT ACCESS:
C     200 RECORDS OF EACH 1000 WORDS (4000 BYTES), I/O WITHOUT FORMAT
C     CONTROLL, IAV IS THE ASSOCIATED VARIABLE
C
C
      WRITE(OUT,10)
   10 FORMAT('1')
      CALL RAMANF(OUT)
      CALL A8FORM(OUT,' MAPLIB ')
      CALL A8FORM(OUT,'-UTILITY')
```

```
      CALL RAMEND(OUT)
CS    RAMANF,A8FORM,RAMEND DIENEN ZUR AUSGABE VON GROSS BUCHSTABEN MIT
CS    UMRAHMUNG
      WRITE(OUT,25)
   25 FORMAT('0',T46,'RELEASE DATE OF MAPLIB-UTILITY: 12.07.70')
C
      CALL PRINTI(5,6)
C     PRINTI  DRUCKT DIE EINGABE AUS
C
      READ (INP,1000,END=999) WORT(1)
      IF(WORT(1).EQ.NEW) GOTO 4
      CALL GETOLD
C     GETOLD  UEBERNIMMT VON DER REGISTER-EINHEIT DEN BISHERIGEN BESTAND
C             IN DIE COMMON-BEREICHE
      CALL PUBLIC('OLD     ')
      IF (WORT(1).NE.OLD) GOTO 3
      GOTO 2
    4 CALL PUBLIC('NEW     ')
    2 READ (INP,1000,END=999) WORT
 1000 FORMAT(20A4)
    3 IF (WORT(1).EQ.ENDE) GOTO 2
      PUNCH=.FALSE.
      LNOC=.FALSE.
      LINK=.FALSE.
      LNEW=.FALSE.
      NOTSTO=.FALSE.
      CALL PAGE
      DO 1 I=1,10
      IF(WORT(1) .EQ. STEUER(I)) GOTO(100,200,300,400,523,600,700,800,
     1 9000,7000),I
    1 CONTINUE
      IKEYER=IKEYER+1
      IF(IKEYER.GE.10) GOTO 2
      WRITE(6,1010) WORT(1),STEUER,OLD,NEW,ENDE
 1010 FORMAT(' *** KEY WORD  *', A4,'*  NOT DEFINED',/
     1 '     PERMISSIBEL KEY WORDS ARE ',(9(' *',A4,'* ')))
      GOTO 2
C
C     DELETE
C
  100 CALL PUBLIC('DELETE  ')
  102 READ(INP,1000,END=999) WORT
      IF(WORT(1).EQ.ENDE) GOTO 102
      CALL INFO(WORT,I,&3)
      IF(WORT(1).EQ.STINFO(7)) GOTO 112
      IF(WORT(1).EQ.STINFO(10))GOTO 113
      IF(WORT(1).EQ.STINFO(9)) GOTO 114
      IKEYER=IKEYER+1
      IF(IKEYER.GT.10) GOTO 102
      WRITE(6,1010) WORT(1),STINFO(7),(STINFO(I),I=9,10),ENDE
      FERROR=.TRUE.
      GOTO 102
  112 CALL DELARG(&999)
      GOTO 102
  113 CALL DELSYS (&999)
      GOTO 102
  114 CALL DELFUN(&999)
```

```
      GOTO 102
C
C  COMMENT
C
  200 CALL PUBLIC('COMMENT ')
      ERKFOL=.TRUE.
      NOTSTO=.TRUE.
      CALL ADDFUN
      ERKFOL=.FALSE.
      GO TO 2
C
C  STORE
C
  300 CALL PUBLIC('STORE   ')
      READ(INP,1000,END=399) WORT
      IF (W8.NE.PASSW(1)) GOTO 398
      IF (ERROR) GOTO 397
      CALL ZEICH($)
      IF($2.NE.DOLLAR)NEWMAS=.TRUE.
      FUNCDA=FUNCDA.AND. .NOT. NEWMAS
      NOTARG=NOTARG.OR.NEWMAS
      $2=DOLLAR
      IF(NEWMAS)CALL PSDAT($,NF1)
      NEWMAS=.FALSE.
      IF(NOTARG) CALL SCRA
      NOTARG=.FALSE.
      CALL PUTOLD
      GO TO 2
  397 CALL FEHLER('ERROR=.TRUE.  ***NOSTORE',24,&2)
  398 WRITE(6  ,3001) WORT
 3001 FORMAT(' *** ERROR ***  PASSWORD ',2A4,'  WRONG  ***NOSTORE')
      GO TO 3
  399 CALL FEHLER('PASSWORD MISSING *** NOSTORE',28,&999)
C
C  ADD
C
  400 NOTSTO=.TRUE.
      NOTARG=.TRUE.
      CALL ADD(WORT,&3,&999)
C
C  INFORMATION
C
  523 CALL PUBLIC('INFO    ')
  500 IF (END) GOTO 999
      READ (INP,1000,END=999) WORT
      IF(WORT(1).EQ. ENDE)GOTO 500
      OUTPUT=.FALSE.
      IS=0
      CALL INFO(WORT,IS,&3)
      IF (IS.NE.0) GOTO (504,504,504,509),IS
      DO 520 IS=1,9
      IF (WORT(1) .EQ. STINFO(IS)) GOTO (501,502,503,504,505,506,507,
     1                                   508,509),IS
  520 CONTINUE
      IF(WORT(1).EQ.STINFO(10)) GOTO 5200
      IF(WORT(1).EQ.STINFO(11)) GOTO 5300
      IF(WORT(1).EQ.STINFO(12)) GOTO 5400
```

```
      IF(.NOT.LINK) GOTO 599
      DO 524 I=1,5
      IF(WORT(1).EQ.STLINK(I)) GOTO (531,532,532,535,536),I
  524 CONTINUE
      WRITE(6,1010) WORT(1),STINFO(1),STINFO(2),STINFO(4),
     1(STINFO(I),I=9,11),STLINK
      FERROR=.TRUE.
      GOTO 3
  531 CALL JCBCDO(INP)
      GOTO 500
  532 READ(INP,1000,END=999) WORT
      IF(I-2)3,533,534
  533 CALL EXEC4(WORT(1))
      GOTO 500
  534 CALL EXEC3(WORT(1))
      GOTO 500
  535 CALL EXEC 5(INP)
      GOTO 500
  536 CALL JCBCD2(INP)
      GOTO 500
  599 WRITE(6  ,1010) WORT(1),STINFO
      GOTO 500
 5400 LNOC=.TRUE.
      GOTO 500
 5200 N=1
      IF(LNEW) N=K$O+1
      IF(N.GT.K$) CALL FEHLER('THERE ARE NO NEW SYSTEM ROUTINES ',
     1 33,&500)
 5202 DO 5203 I=N,K$
 5203 CALL FILOUT(NF1,NRECSA(I),NRECSE(I),'SYSTEM  ROUTINE',15)
      GOTO 500
 5300 LNEW=.TRUE.
      GOTO 500
C
C  .ALL
C
  501 IF (PUNCH) GOTO 502
      GOTO 503
C
C  .LIST
C
  502 IS=14
      GO TO 504
C
C  .MAP
C
  503 GOTO 505
C
C  .MASTER
C
  504 IF (ERROR) CALL FEHLER('THERE IS NO NEW MASTER',22,&500)
      IF (.NOT.NEWMAS) GOTO 5041
      CALL ZEICH($)
      CALL PSDAT($,NF2)
      FUNCDA=.FALSE.
      NEWMAS=.FALSE.
      IF(JEIN.EQ.0) CALL FEHLER('THERE IS NO MASTER',18,&500)
```

```
 5041 CONTINUE
      IF (.NOT.LINK .OR. IS.LT.4) GOTO 5043
      OUTPUT=.TRUE.
      GO TO 510
 5043 IF (IS.LT.4) GOTO 510
      CALL FILOUT(NF2,N$$A,N$$E,'MASTER  FUNCTION',16)
      N=1
      IF(LNEW) N=KSO+1
      IF(N.GT.KS) CALL FEHLER('NO NEW MATERIAL MASTER FUNCTIONS',32,
     1 &5070)
      DO 5060 I=N,KS
 5060 CALL FILOUT(NF2,NRSTOA(I),NRSTOE(I),
     1'MATERIAL  MASTER  FUNCTION',26)
 5070 CONTINUE
      N=1
      IF(LNEW) N=KEO+1
      IF(N.GT.KE) CALL FEHLER('NO NEW PROPERTY MASTER FUNCTIONS',32,
     1 &5071      )
      DO 5072 I=N,KE
 5072 CALL FILOUT(NF2,NREIGA(I),NREIGE(I),'PROPERTY  MASTER  FUNCTION',
     1 26)
 5071 CONTINUE
      IF (IS.EQ.4) GOTO 500
      GO TO 509
C
C  .MATERIALS
C
  505 IF (KS.EQ.0) CALL FEHLER('NO MATERIALS IN CATALOGUE',25,&5051)
      DO 550 I=1,KS
      IF(MOD((I-1),50) .EQ. 0) WRITE(OUT,1500)
 1500 FORMAT('1** MATERIALS'//'OSYMBOL',T11,'MATERIAL'//)
  550 WRITE(OUT,1501) KATSTO(I),(ERKSTO(J,I),J=1,17)
 1501 FORMAT(' ',A4,T10,18A4)
 5051 IF (IS.EQ.5) GOTO 500
C
C  .PROPERTIES
C
  506 IF (KE.EQ.0) CALL FEHLER('NO PROPERTIES IN CATALOGUE',26,&5061)
      DO 560 I=1,KE
      IF (MOD((I-1),50) .EQ. 0) WRITE(OUT,1600)
 1600 FORMAT('1** PROPERTIES '//'OSYMBOL',T11,'PROPERTY',T64,'UNIT'//)
  560 WRITE(OUT,1601) KATEIG(I),(ERKEIG(J,I),J=1,17)
 1601 FORMAT(' ',A2,T10,18A4)
 5061 IF (IS.EQ.6) GOTO 500
C
C   .PARAMETERS
C
  507 IF(KA.EQ.0) CALL FEHLER('NO PARAMETERS IN CATALOGUE',26,&5074)
      DO 570 I=1,KA
      IF (MOD((I-1),50) .EQ. 0) WRITE(OUT,1700)
 1700 FORMAT('1** PARAMETERS'//'OSYMBOL',T10,' PARAMETER',T64,'UNIT'//)
  570 WRITE(OUT,1701) (KATARG(J,I),J=1,3),(ERKARG(J,I),J=1,17)
 1701 FORMAT(' ',3A2,T10,18A4)
 5074 IF (IS.EQ.7) GOTO 500
C
C  .TABLE
C
```

```
  508 CALL TABWRT
      IF (IS.EQ.8 .OR. IS.EQ.3) GOTO 500
      GO TO 502
C
C  .FUNCTION
C
  509 IF (NRF11.EQ.0) CALL FEHLER('THERE ARE NO FUNCTIONS',22,&500)
      IF (IS.NE.9 .AND. IS.NE.14 .AND. IS.NE.1) GOTO 513
      GOTO 701
C
C  ..$$$$$$
C
  510 CONTINUE
      IF (IS.EQ.1) OUTPUT=.TRUE.
      IF(.NOT.OUTPUT) GOTO 5105
      CALL F11TO9
                  CALL FILOUT(NDA,N$$A,N$$E, 'MASTER  FUNCTION',16)
 5105 CONTINUE
      IF (IS.EQ.1) GOTO 500
C
C  ..AB$$$$
C
      IF (IS.EQ.2) OUTPUT=.TRUE.
      IF(.NOT.OUTPUT) GOTO 5102
      CALL LSTCON(WERT(2),KATSTO,2,1,I,KS,&5130)
      CALL F11TO9
      CALL FILOUT(NDA,NRSTOA(I),NRSTOE(I),'MATERIAL  MASTER  FUNCTION',
     1 26)
 5102 CONTINUE
      IF (IS.EQ.2) GOTO 500
C
C  ..$$ABCD
C
      IF (IS.EQ.3) OUTPUT=.TRUE.
      IF(.NOT.OUTPUT) GOTO 5103
      CALL LSTCON(EIGEN(2),KATEIG,1,1,I,KE,&5130)
      CALL F11TO9
      CALL FILOUT(NDA,NREIGA(I),NREIGE(I),'PROPERTY  MASTER  FUNCTION',
     1 26)
 5103 CONTINUE
      IF (IS.EQ.14) GO TO 509
      GO TO 500
C
C  AUSGABE EINER EINZIGEN FUNKTION
C
  513 CALL KATCON(EIG,STO,EIGEN(2),STOFF,1,J,JEIN,&5130)
      NF=NDA
      CALL F11TO9
 5131 CALL FILOUT(NF ,NRECA(J),NRECE(J),'MATERIAL  PROPERTY  FUNCTION',2
     18)
      IF(IS.EQ.9  .OR.  IS.EQ.14) GOTO 702
      GO TO 500
 5130 EIGEN(1)=EIGEN(2)
      EIGEN(2)=EIGEN(3)
      EIGEN(3)=EIGEN(4)
      CALL LSTCON(EIGEN,SYSTEM,3,1,J,K$,&5330)
      CALL F11TO9
```

```
 5331 CALL FILCUT(NDA,NRECSA(J),NRECSE(J),'SYSTEM  ROUTINE',15)
      GOTO 500
 533C WRITE(CUT,1513) EIGEN(1),EIGEN(2),EIGEN(3)
 1513 FORMAT('0  FUNCTION  ',3A2,'  NOT FOUND')
      GOTO 500
C
C  SOURCE
C
  6CC PUNCH=.TRUE.
      CALL PUBLIC('SOURCE  ')
      GOTO 500
C
C  LINK
C
  7CC LINK=.TRUE.
      CALL PUBLIC('LINK    ')
      PUNCH=.TRUE.
      GO TO 500
  7C1 NF=NF1
  7C3 N=1
      IF(LNEW)N=JEINO+1
      IF(N.GT.JEIN)CALL FEHLER('THERE ARE NO NEW FUNCTIONS',
     1 26,&7C5  )
      DO 7C2 J=N,JEIN
      GO TO 5131
  7C2 CONTINUE
  7C5 IF(IS.EQ.14) GOTO 52CC
      GO TO 500
C
C     UNIT
C
  8CC CALL $UNIT(&999)
      GOTO 2
C
C     NAME
C
9CCCC CONTINUE
      CALL $NAME(INP,OUT,NEWMAS,&999)
      GOTO 2
C
C    COPY
C
 7CCC CALL PUBLIC('COPY    ')
      CALL COPY(WORT,&3,&999)
      GOTO 2
C
C     STOP
C
  999 CALL PUBLIC('STOP    ')
 992C IF(.NOT.FERROR .AND. .NOT. ERROR)WRITE(6,9921)
 9921 FORMAT('0 NO ERRORS WERE NOTICED IN THIS JOB BY MAPLIB-UTILITY')
      IF(ERROR) CALL FEHLER('AT LEAST 1 SYSTEM-ERROR HAS BEEN NOTICED; C
     1ALL THE MAPLIB-ENGINEER',66,&9922)
      IF(FERROR) CALL FEHLER('AT LEAST 1 INPUT-ERROR HAS BEEN NOTICED',
     1  39,&9922)
 9922 STOP
      END
```

```
      SUBROUTINE GETOLD
CA    GETOLD
C
CB    GETOLD LIEST VON DER REGISTER-EINHEIT NF3 DEN BISHERIGEN MAPLIB-
CB    BESTAND IN DIE COMMON VARIABLEN EIN.
CB    UND INITIALISIERT DIE ANFANGSVARIABLEN JEINO,KEO,...
CB    DIE EIN-AUSBABE ERFOLGT FORMATGEBUNDEN UM LESBAR ZU SEIN.
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTOA(100),NRSTOE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON/SDIM2/ EIG(500),NARG(500),KATEIG(100),SYNEIG(20),KATARG
     1(3,50),SYSTEM(3,50)
      INTEGER*2 EIG,NARG,KATEIG,SYNEIG,KATARG,SYSTEM
      COMMON/SDIM1/ LSYN(500),LERL(4000),FIL(80,100)
      LOGICAL*1LSYN,LERL,FIL
      COMMON/SKCNT8/RELDAT,RELUHR,DAT,UHR
      REAL*8 RELDAT,RELUHR,DAT,UHR
      COMMON/SKCNT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      COMMON/SKCNT2/ NAMAX,NAMAXO
      INTEGER*2 NAMAX,NAMAXO
      COMMON/SKCNT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1    ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,  FUNCDA,NEWMAS,PUNCH
C
      REWIND NF3
      READ(NF3,107,END=20,ERR=10)RELDAT,RELUHR
  107 FORMAT(1X,2A8)
      READ(NF3,100,END=20,ERR=10) JEIN,KE,KS,KA,KY,NAMAX,NRF11,N$$A,
     1 N$$E,K$
  100 FORMAT(1X,9I8,I7)
      JEINO=JEIN
      KEO=KE
      KSO=KS
      KAO=KA
      KYO=KY
      NAMAXO=NAMAX
      K$O=K$
      IF(JEIN.GT.JEMAX)CALL FEHLER('JEIN.GT.JEMAX *GETOLD',21,&11)
      IF(KE  .GT.KEMAX)CALL FEHLER('KE.GT.KEMAX   *GETOLD',21,&11)
      IF(KS  .GT.KSMAX)CALL FEHLER('KS.GT.KYMAX   *GETOLD',21,&11)
      IF(KY  .GT.KYMAX)CALL FEHLER('KY.GT.KYMAX   *GETOLD',21,&11)
      IF(K$  .GT.K$MAX)CALL FEHLER('K$.GT.K$MAX   *GETOLD',21,&11)
      IF(NAMAX.GT. 48) CALL FEHLER('NAMAX.GT.48   *GETOLD',21,&11)
      IF(K$.LT.1) GOTO 1021
      DO 1022 I=1,K$
 1022 READ(NF3,106,END=20,ERR=10)(SYSTEM(J,I),J=1,3),NRECSA(I),NRECSE(I)
  106 FORMAT( 1X,3A2,3I10)
 1021 CONTINUE
      IF(JEIN.LT.1) GOTO 21
      DO 1 I=1,JEIN
    1 READ(NF3,101,END=20,ERR=10) EIG(I),STO(I),(ARG(J,I),J=1,6),
```

```
     1NRECA(I),NRECE(I),NARG(I),LSYN(I)
  101 FORMAT(1X,A2,7A4,3I8,L1)
   21 IF(KE.LT.1) GOTO 31
      DO 2 I=1,KE
    2 READ(NF3,102,END=20,ERR=10)KATEIG(I),(ERKEIG(J,I),J=1,17)
     1,NREIGA(I),NREIGE(I)
  102 FORMAT(1X,A2,5X,17A4/(1X,8I9))
   31 IF(KS.LT.1) GOTO 41
      DO 3 I=1,KS
    3 READ(NF3,103,END=20,ERR=10)KATSTO(I),(ERKSTO(J,I),J=1,17)
     1,NRSTOA(I),NRSTOE(I)
  103 FORMAT(1X,A4,3X,17A4/(1X,8I9))
   41 IF(KY.LT.1) GOTO 51
      DO 4 I=1,KY
    4 READ(NF3,104,END=20,ERR=10) SYNEIG(I),SYNSTO(I),JSYN(I)
  104 FORMAT(1X,A2,A4,I10)
   51 IF(KA.LT.1) GOTO 61
      DO 5 I=1,KA
    5 READ(NF3,105,END=20,ERR=10)(KATARG(J,I),J=1,3),(ERKARG(J,I),J=1,17
     1)
  105 FORMAT(1X,3A2,1X,17A4)
   61 CONTINUE
      RETURN
   20 CALL FEHLER('END OF FILE TOO EARLY',21,&23)
   10 CALL FEHLER('I/O-ERROR',9,&23)
   11 CALL FEHLER('DIMENSION-ERROR',15,&23)
   23 CALL FEHLER('IN GETOLD',9,&22)
   22 ERROR=.TRUE.
      GOTO 61
      ENTRY PUTOLD
CN    PUTOLD
CB    FIXIERUNG DES MAPLIB-BESTANDES AUF DER REGISTER-EINHEIT FUER
CB    SPAETERE JOBS; WIRD NACH KEY STORE AUFGERUFEN.
      REWIND NF3
      RELDAT=DAT
      RELUHR=UHR
      WRITE(NF3,107) RELDAT,RELUHR
      WRITE(NF3,100)JEIN,KE,KS,KA,KY,NAMAX,NRF11,N$$A,N$$E
     1 ,K$
      IF(K$.LT.1)GOTO2021
      DO 2022 I=1,K$
 2022 WRITE(NF3,106) (SYSTEM(J,I),J=1,3),NRECSA(I),NRECSE(I)
 2021 CONTINUE
      IF(JEIN.LT.1) GOTO 201
      DO 211 I=1,JEIN
  211 WRITE(NF3,101) EIG(I),STO(I),(ARG(J,I),J=1,6),NRECA(I),NRECE(I),
     1NARG(I),LSYN(I)
  201 IF(KE.LT.1) GOTO 202
      DO 212 I=1,KE
  212 WRITE(NF3,102) KATEIG(I),(ERKEIG(J,I),J=1,17),NREIGA(I),NREIGE(I)
  202 IF(KS.LT.1) GOTO 203
      DO 213 I=1,KS
  213 WRITE(NF3,103) KATSTO(I),(ERKSTO(J,I),J=1,17),NRSTOA(I),NRSTOE(I)
  203 IF(KY.LT.1) GOTO 204
      DO 214 I=1,KY
  214 WRITE(NF3,104) SYNEIG(I),SYNSTO(I),JSYN(I)
  204 IF(KA.LT.1) GOTO 61
```

```
      DO 215 I=1,KA
  215 WRITE(NF3,105)(KATARG(J,I),J=1,3),(ERKARG(J,I),J=1,17)
      GOTO 61
      END
```

```
      SUBROUTINE ADD(WORT,*,*)
CN    ADD
CB    DIESE ROUTINE WIRD VON MAPLIB NACH DEM HAUPTSCHLUESSELWORT 'ADD '
CB    AUFGERUFEN.
CB    SIE RUFT IHRERSEITS JE NACH DEM UNTERSCHLUESSELWORT  .FUN  ODER
CB    .SYS  DIE ROUTINE  ADDFUN ODER ADDSYS AUF
CP    WORT  BEIM RUECKSPRUNG MIT RETURN1 ENTHAELT WORT DAS NAECHSTE
CP          SCHLUESSELWORT, DAS WEDER .FUN NOCH . SYS IST
C
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      INTEGER WORT(2),$$(2)/'.FUN','.SYS'/
     1,END/'END '/
      CALL PUBLIC('ADD     ')
      CALL CREATE
C
CB    CREATE IST ENTRY IN GETREC; BEWIRKT DEFINE FILE
C
      CALL FIND11(NRF11,NF1)
C
CB    AM ANFANG SETZT ADD DEN LESEKOPF DER EINHEIT NF1 AUF DEN ERSTEN
CB    RECORD NACH DEM LETZTEN END-STATEMENT MIT DER ROUTINE FIND11
C
  100 READ(INP,1000,END=999)WORT
 1000 FORMAT(3A4)
      IF(WORT(1).EQ.END) GOTO 100
      DO 99 I=1,2
      IF(WORT(1).EQ.$$(I)) GOTO(1,2),I
   99 CONTINUE
      CALL END11(NF1)
C
CB    AM ENDE WIRD MIT END11 AUF NF1 DER RECORD 'END OF FILE' GESCHRIE-
CB    BEN
C
      RETURN1
    1 CALL ADDFUN
      GOTO 100
    2 CALL ADDSYS
      GOTO 100
  999 CALL END11(NF1)
CP    RETURN2  BEI END OF FILE ON UNIT INP
      RETURN 2
      END
```

```
      SUBROUTINE ADDFUN
CN    ADDFUN
C
CB    DIESE ROUTINE WIRD NACH DEN SCHLUESSELWORTEN ADD/.FUN  VON ADD UND
CB    NACH  COMM  VON MAPLIB MIT ERKFOL=.TRUE. AUFGERUFEN
CB    ZUSAMMEN MIT  FUNANA UND KATALG WERDEN HIER DIE NEU EINGEBRACHTEN
CB    FUNKTIONEN REGISTRIERT, UEBERARBEITET UND AUF NF1 AUSGEGEBEN.
C
CI    ADDFUN ERWARTET SEINE EINGABE VON DIRECT-ACCES; ADDFUN LIEST
CI    JEDOCH NICHT SELBST, SONDERN BEKOMMT DIE EINGABE KARTEN(RECORD)-
CI    WEISE MIT GETREC BZW. BYTE-WEISE MIT GETFIL
CI    DIESE LESEN NICHT KARTEN-WEISE SONDERN JEWEILS BLOECKE ZU 50
CI    KARTEN
CI    DAS DURCHSUCHEN DER EINGABE NACH BESTIMMTEN TEXTEN GESCHIEHT MIT
CI    HILFE DER ROUTINE SUCHR
C
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTOA(100),NRSTOE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON/SDIM2/ EIG(500),NARG(500),KATEIG(100),SYNEIG(20),KATARG
     1(3,50),SYSTEM(3,50)
      INTEGER*2 EIG,NARG,KATEIG,SYNEIG,KATARG,SYSTEM
      COMMON/SDIM1/ LSYN(500),LERL(4000),FIL(80,100)
      LOGICAL*1LSYN,LERL,FIL
      COMMON/SKCNT8/RELDAT,RELUHR,DDAT,DZEIT
      REAL*8 RELDAT,RELUHR,DDAT,DZEIT
      COMMON/SKCNT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KF,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      COMMON/SKCNT2/ NAMAX,NAMAXO
      INTEGER*2 NAMAX,NAMAXO
      COMMON/SKCNT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1    ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,   FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 LARG$(24,500),DOLLAR/'$'/
      EQUIVALENCE(ARG(1,1),LARG$(1,1))
C
      REAL*8 FK,FKTIO/'FUNCTION'/,ENT/'ENTRY   '/
      INTEGER*2 NARG1,A2,EIGEN,SEIG,ARGAKT(3,48),SST(2),NAMENT(3,6),
     1 EIGA
      INTEGER SSTO,STOFF,IRENT(6)
     1,A4(20)
      LOGICAL*1 LREND,LOR,LOF,LORPRO,LOZ,LOZPRO,LOGPRO,LNAM,LDAT,LA(6),
     1           LOG,LAA(80),LKKD,LSSD,LOS,GETFIL,NAME(30),
     2           NONAME( 4)/'N.N.'/         ,DAT(10),LS(4),LAK(6),NOTCAT
     2,LO$BPR,LO$B,LO$T,NT(5)
     3,NRET(5),LITER(66),LOL
      INTEGER*2 DUM(3,2)/'I     DUMMY '/
      EQUIVALENCE(A4(1),A2,EIGEN,LAA(1),LA(1)),(SST(1),SSTO),
     1(LS(1),STOFF),(ARGAKT(1,1),LAK(1))
      CALL FSPIE
      CALL FILFIL(INP,NDIMAX,FIL,LERL)
C
```

```
CB     FILFIL SCHREIBT EINGABE AUF DIRECT ACCESS
C
C
CB     INITIALISIERUNG
       FUNCDA=.FALSE.
       IRMAX=IAV-2
       IREND=IRMAX
       NF11=NF1
       NF1=0
C
CB     NF11 = ZWISCHENSPEICHER VON NF1. NF1=0 IST STEUERANWEISUNG FUER
CB     KATALG UND BEWIRKT, DASS IN KATALO NICHT AUF NF1 GESCHRIEBEN WIRD
C
       KE1=KE
       KS1=KS
       KA1=KA
       KY1=KY
       JEIN1=JEIN
       IF (ERKFEL) GOTO 1003
C
CB     IF(ERKFEL): NUR ANALYSE VON KOMMENTARKARTEN NACH COMMENT
C
       CALL SUCHR('FUNCTION',8,7,1,IRMAX,IREND,&999)
       IREND=IREND-1
       LDAT=.FALSE.
       LNAM=.FALSE.
       LCL=.FALSE.
       LCGPRC=.FALSE.
       LCZPRC=.FALSE.
       LCRPRC=.FALSE.
       LC$PPR=.FALSE.
       IF(IREND.LT.1) GOTO 1100
C
CB     IF(IREND.LT.1) -> KEIN PROLOG VORHANDEN
C
CB     PROLOG  VERARBEITUNG
C
       CALL FILOUT(NDA,1,IREND,'PROLOG  INPUT',13)
C      AUSGABE
C
       CALL SUCHR('C$B',3,1,1,IREND,I,&1033)
       LC$PPR=.TRUE.
       LERL(I)=.TRUE.
 1033  CONTINUE
       CALL SUCHR('CL',2,1,1,IREND,J,&1031)
       LERL(J)=.TRUE.
       LCL=.TRUE.
       DO 1032 I=1,66
 1032  LITER(I)=GETFIL(I+6,J)
 1031  CONTINUE
       CALL SUCHR('C$R',3,1,1,IREND,I,&1001)
       LERL(I)=.TRUE.
       LCRPRC=.TRUE.
 1001  CALL SUCHR('C$Z',3,1,1,IREND,I,&1002)
       LERL(I)=.TRUE.
       LCZPRC =.TRUE.
 1002  CALL SUCHR('C$G',3,1,1,IREND,I,&1003)
```

```
      LERL(I)=.TRUE.
      LCGPRC =.TRUE.
 1003 J=1
C
CB    HIER BEGINNT DIE VERARBEITUNG NACH COMMENT
C
      IF(ERKFOL)CALL FILCUT(NCA,1,IREND,'COMMENT  INPUT',14)
 1004 CALL SUCHR('C$P',3,1,J,IREND,J,&1005)
      LERL(J)=.TRUE.
      LA(1)= GETFIL(7,J)
      LA(2)= GETFIL(8,J)
      CALL KATALC(KATEIC,1,KE,KEMAX,ERKEIC,EIGEN,'$P',J+1,-1,OUT,NF1,
     1&999)
      LERL(J+1)=.TRUE.
      J=J+2
      GOTO 1004
 1005 J=1
 1006 CALL SUCHR('C$M',3,1,J,IREND,J,&1007)
      LERL(J)=.TRUE.
      DO 1008 I=1,4
 1008 LS(1)= GETFIL(I+6,J)
      CALL KATALC(KATSTC,2,KS,KSMAX,ERKSTC,STOFF,'$M',J+1,-1,OUT,NF1,
     1&999)
      LERL(J+1)=.TRUE.
      J=J+2
      GOTO 1006
 1007 J=1
 1009 CALL SUCHR('CP', 2,1,J,IREND,J,&1010)
      LERL(J)=.TRUE.
      LERL(J+1)=.TRUE.
      DO 1011 I=1,6
 1011 LAK(I)=GETFIL(I+6,J)
      CALL KATALC(KATARG,3,KA,KAMAX,ERKARG,LAK,'CP',J,J+1,OUT,NF1,&999)
      J=J+2
      GOTO 1009
 1010 IF (ERKFOL) GOTO 1030
      CALL SUCHR('CA', 2,1,1,IREND,J,&1020)
      LERL(J)=.TRUE.
      LNAM=.TRUE.
      DO 1015 I=1,30
 1015 NAME(I)=GETFIL(I+6,J)
 1020 CALL SUCHR('CD',2,1,1,IREND,J,&1030)
      LERL(J)=.TRUE.
      LDAT=.TRUE.
      DO 1021 I=1,10
 1021 DAT  (I)=GETFIL(I+6,J)
 1030 NOTCAT=.TRUE.
      DO 1100 I=1,IREND
      IF(LERL(I)) GOTO 1100
      IF (NOTCAT) WRITE(OUT,1012)
 1012 FORMAT('0NOT CATALOGIZED STATEMENTS:'//)
      NOTCAT=.FALSE.
      CALL GETREC(A4,I)
      WRITE(OUT,1014) A4
 1014 FORMAT(' ',20A4)
 1100 CONTINUE
CB
```

```
CB     PROLOG  VERARBEITUNG  ENDE
CB
       NF1=NF11
C
CB     RUECKSPEICHERUNG VON NF1
C
       IF (ERKFEL) RETURN
C
     3 LREND=.FALSE.
CV     LREND =.TRUE., WENN DER LETZTE EINGABEBLOCK VERARBEITET WIRD
       IREND=IREND-1
       GO TO 100
     5 JEIN=JEIN1
C
CB     BEI 5 WIRD FORTGESETZT, WENN EIN EINGABEFEHLER DIE AUFNAHME DER
CB     FUNKTION UNMOEGLICH MACHT
CB     DIE REGISTER WERDEN AUF DEN ZUSTAND BEI BEGINN DER VERARBEITUNG
CB     DIESER FUNKTION ZURUECKGESETZT
       KE=KE1
       KA=KA1
       KY=KY1
       KS=KS1
C
    10 KE1=KE
CB     FESTHALTEN DES AUSGANGSZUSTANDES
       KY1=KY
       KA1=KA
       KS1=KS
       IF(LREND) GOTO 1000
   100 JEIN1=JEIN
       IRFUN1=IREND + 2
       CALL FROMRC(IRFUN1)
CB     FROMRC IST ENTRY IN GETREC; HIER WIRD DIE KLEINSTE NOCH JEMALS
CB     BENOETIGTE RECORD-NUMMER ANGEGEBEN; DIES DIENT ZUR BESCHLEUNIGUNG
CB     DER EINGABE IN GETREC
       J= IRFUN1
       NRF11O=NRF11+1
       I=IRFUN1+1
       IF(I.GE.IRMAX) GOTO 1000
     6 CALL SUCHR('END      ',8,7,I,IRMAX,IREND,&199)
       CALL SUCHR('C$C',3,1,IREND-1,IREND-1,I,&198)
       I=IREND+1
       GOTO 6
   198 I=IRFUN1
   195 CALL SUCHR('FUNCTION',8,7,I,IREND,IRFUN1,&197)
   196 IF(IRFUN1.EQ.J) GOTO 2
       CALL FROMRC(IRFUN1)
     8 CALL FILOUT(NDA,J,IRFUN1-1,'NOT CATALOGIZED STATEMENTS',26)
       FERROR=.TRUE.
       GOTO 2
   199 LREND=.TRUE.
       CALL FILOUT(NDA,J,IRMAX,'NOT CATALOGIZED STATEMENTS',26)
       FERROR=.TRUE.
       GOTO 1000
   197 CALL FEHLER('NO FUNCTION STATEMENT FOUND',   27,&199)
     2 IREND=IREND-1
       DO 11 I=2,6
```

```
   11 IRENT(I)=0
      CALL FILCUT(NDA,IRFUN1,IREND+1,'FUNCTION  INPUT',15)
      IRENT(1)=IRFUN1
      CALL NRRETU(IRFUN1,IREND,IRET,NRET,&5)
C
CB    FESTSTELLUNG DER STATEMENT-NR NRET VOR RETURN
C
      DO 12 I=1,6
      IF(I.GT.1) IRENT(I)=IRENT(I-1)
   13 CALL SUCHR('ENTRY',5,7,IRENT(I)+1,IREND,IRENT(I),&14)
   12 CONTINUE
      CALL FEHLER('THE FUNCTION HAS MORE THAN SIX ENTRIES',38,&14)
   14 IENT=I-1
C
CB    FESTSTELLUNG DER ZAHL DER ENTRIES IENT
C
      NRENT= 0
C
CV    NRENT = NR. DES ENTRY-BLOCKS, DER VERARBEITET WIRD
CV    BEI NRENT=0  WERDEN DIE KARTEN VOR DEM ERSTEN ENTRY VERARBEITET
C
      FK=FKTIO
   30 IF(NRENT.GT.0) IRFUN1=IRENT(NRENT)
      IREND1=IREND
      IF(NRENT.LT.IENT) IREND1 = IRENT(NRENT+1)-1
      IF(NRENT.GT.0) FK=ENT
      IF(FK.EQ.ENT) GOTO 335
      LOR= LORPRO
      LO$B=LO$BPR
      LO$T=.FALSE.
      LOZ= LOZPRO
      LOG= LOGPRO
      LOF= .FALSE.
      LOS=.FALSE.
      LKKO=.FALSE.
      LSSO=.FALSE.
      NAMIN=100
      CALL SUCHR('O$N',3,1,IRFUN1,IREND,I,&4370)
      NAMIN=NUMBRI(GETFIL(7,I),GETFIL(8,I))
 4370 CONTINUE
      CALL SUCHR('O$B',3,1,IRFUN1+1,IREND,I,&52)
      LO$B=.TRUE.
      LERL(I)=.TRUE.
   52 CALL SUCHR('O$T',3,1,IRFUN1,IREND,IRTZ,&53)
      LO$T=.TRUE.
      LERL(IRTZ)=.TRUE.
      IRTZ=IRTZ+1
   53 CONTINUE
      CALL SUCHR('O$R',3,1,IRFUN1,IREND,I,&31)
      LERL(I)=.TRUE.
      LOR=.TRUE.
   31 CONTINUE
      CALL SUCHR('O$Z',3,1,IRFUN1,IREND,I,&33)
      LERL(I)=.TRUE.
      LOZ=.TRUE.
   33 CONTINUE
      CALL SUCHR('O$G',3,1,IRFUN1,IREND,I,&34)
```

```
      LERL(I)=.TRUE.
      LCG=.TRUE.
   34 CALL SUCHR('C$S',3,1,IRFUN1,IREND,IRCS,&32)
      LERL(IRCS)=.TRUE.
      LCS=.TRUE.
      DO 36 I=1,2
   36 LA(I)=GETFIL(I+6,IRCS)
      EIGA=EIGEN
      DO 336 I=1,4
      LS(I)=GETFIL(I+8,IRCS)
  336 LA(I+2)=LS(I)
      CALL KATCON(SYNEIG,SYNSTO, EIGEN,STOFF, 1,JY,KY,&32)
      CALL FEHLER('THE NEW FUNCTION-NAME HAS BEEN STILL USED AS SYNONYM'
     1,52,&5)
   32 CALL SUCHR('C$F',3,1,IRFUN1,IREND,IRCF,&9934)
      LCF=.TRUE.
      GOTO 54
 9934 CALL SUCHR('      ',6,1,IRFUN1+1,IREND,IRCF,&9935)
      GOTO 54
 9935 CALL FEHLER('C$F BEFORE THE FIRST EXECUTABLE STATEMENT MISSING',
     149,&5)
   54 IF(LC$T) GOTO 55
      IRTZ=IRCF
      IF(LCF) IRTZ=IRTZ+1
   55 IF(IRET.EQ.IRTZ) CALL FEHLER('THE FIRST STATEMENT AFTER THE TEST Z
     1ONE IS A RETURN; THIS IS NOT ALLOWED',72,&5)
      IF(IRTZ.LT.IRCF) CALL FEHLER('C$T BEFORE C$F',14,&5)
      CALL NRTEST(NT,NRET,IRFUN1,IREND,IRTZ,&5)
      IF(.NOT.LC$T) GOTO 35
      CALL SUCHR('$TEST$',6,14,IRFUN1+1,IRTZ,I,&35)
      LC$B=.FALSE.
      CALL FEHLER('THE TEST ZONE IS ALREADY BYPASSED',33,&35)
  335 CALL FRCNFC(IRFUN1)
   35 CONTINUE
      NARG1=1
CB    NARG1=1  DIENT ALS STEUERPARAMETER FUER FUNANA
      CALL FUNANA(FK,SEIG,SSTO,NARG1,ARGAKT,IRFUN1,ARG(1,JEIN+1),
     1 NF1,NAMIN,&5)
C
CB    FUNANA VERARBEITET DAS FUNCTION ODER ENTRY-STATEMENT
C
      IF (FK.NE.ENT) GOTO 49
      NAMENT(1,NRENT) = SEIG
      NAMENT(2,NRENT) = SST(1)
      NAMENT(3,NRENT) = SST(2)
      GO TO 312
   49 CALL SUCHR('CN',2,1,IRFUN1,IREND,I,&46)
      I=I-1
      J=I+2
      DO 47 K=I,J,2
   47 CALL SUCHR('C                                    ',30,1,K,K,N,&46)
      GOTO 48
   46 WRITE(NF1,300)
      NRF11=NRF11+3
  300 FORMAT('C'/'CN    ***  M A P L I B  ***   FUNCTION'/'C')
   48 CALL SUCHR('CD',2,1,IRFUN1,IREND,I,&301)
      GOTO 316
```

```
  301 IF (LDAT) GO TO 311
      WRITE (NF1,313) DDAT
  313 FORMAT('CD    ',A8)
      GO TO 314
  311 WRITE(NF1,303) DAT
  303 FORMAT('CD    ',10A1)
  314 NRF11 = NRF11+1
  316 CONTINUE
      CALL SUCHR('CA',2,1,IRFUN1,IREND,I,&304)
      GOTO 317
  309 WRITE(NF1,306) NAME
      GO TO 315
  304 IF (LNAM) GO TO 309
      WRITE(NF1,306) NONAME
  315 NRF11 = NRF11+1
  306 FORMAT('CA',4X,30A1)
  317 CALL SUCHR('CL',2,1,IRFUN1,IREND,I,&318)
      CALL SUCHR('CL-CARD MISSING ',16,1,I,I,I,&319)
      LERL(I)=.TRUE.
  318 IF(LCL) GOTO 321
      WRITE(NF1,322)
  322 FORMAT('CL-CARD MISSING ')
      NRF11=NRF11+1
      WRITE(6,323)
  323 FORMAT('0*** WARNING. CL-CARD MISSING ')
      GOTO 319
  321 WRITE(NF1,320) LITER
  320 FORMAT('CL',4X,66A1)
      NRF11=NRF11+1
  319 CONTINUE
      IF(LCS) GOTO 37
      EIGEN= SEIG
      STOFF= SSTO
      EIGA=EIGEN
   37 CALL KATCON(EIG,STO,EIGEN,STOFF, 1,JDO,JEIN  ,&38)
C
CB    JDO   IST DIE NUMMER DER NEUEN FUNCTION
CB    BEI REPLACEN IST  JDO  DIE ALTE NUMMER
CB    BEI NEUEINGABE  IST  JDO=JEIN+1;  SIEHE NR. 38
C
      LKKO=.TRUE.
      IF(.NOT.LCR) GOTO 42
      IF(.NOT.LSYN(JDO)) GOTO 56
      I=NRSYN(JDO)
      IF(.NOT.LCS .OR. (SEIG.NE.SYNEIG(I) .OR. SSTO.NE.SYNSTO(I)))
     1 CALL DELSYN(JDO)
   56 DO 51 I=1,6
   51 ARG(I,JDO)=ARG(I,JEIN+1)
      GOTO 40
   38 JDO=JEIN+1
      EIG(JDO )=EIGEN
      STO(JDO )=STOFF
   40 LSYN(JDO)=LCS
      IF (.NOT. LCS) GOTO 42
   39 CALL KATCON(SYNEIG,SYNSTO,SEIG,SSTO,1,JY,KY,&41)
      LSSO=.TRUE.
      IF(JSYN(JY).NE.JDO) CALL FEHLER('THIS SYNONYM BELONGS TO AN OTHER
```

```
     1FUNCTION',40,&5)
      GOTO 42
   41 KY=KY+1
      JY=KY
      IF(KY.GT.KYMAX) CALL FEHLER('KY.GT.KYMAX',11,&999)
      SYMEIG(KY)=SEIG
      SYMSTO(KY)=SSTO
      JSYM(KY)  =JDO
   42 LKKO=LSSD .OR. LKKO
      IF(.NOT.LOR.AND.          LKKO) CALL FEHLER('DOUBLE INPUT AND NO RE
     1PLACE-OPTION',34,&5)
      CALL KATALO(KATEIG,1,KE,KEMAX,ERKEIG,EIGEN,'$P',IRFUN1,IREND,OUT,
     1 NF1,&999)
      CALL KATALO(KATSTO,2,KS,KSMAX,ERKSTO,STOFF,'$M',IRFUN1,IREND,OUT,
     1NF1,&999)
      DO 110 J=1,NARG1
      CALL KATALO(KATARG,3,KA,KAMAX,ERKARG,ARGAKT(1,J),'OP',IRFUN1,
     1  IREND,OUT,NF1,&999)
      DO 204 I=1,3
      IF(ARGAKT(I,J) .NE. DUM(I,1)) GOTO 110
  204 CONTINUE
      CALL FEHLER('THE PARAMETER-NAME ''I '' IS NOT ALLOWED',38,&5)
  110 CONTINUE
      I=1
      DO 201 J=1,3
      IF(ARGAKT(J,I) .NE. DUM(J,2)) GOTO 202
  201 CONTINUE
      GOTO 203
  202 CONTINUE
      I= 0
  203 NARG1=NARG1-I
      NARG(JDO)=NARG1
      IF(NARG1.GT.NAMAX)NAMAX=NARG1
      IF(.NOT.LOS) GO TO 307
      WRITE(NF1,308) LA
      NREL1 = NREL1+1
      JSYM(JY)=JDO
      LSYM(JDO)=.TRUE.
  308 FORMAT('C$S',3X,6A1)
  307 CONTINUE
  312 DO 372 K=IRFUN1,IREND1
      IF(LERL(K)) GOTO 372
      IF(K.NE.IROF) GOTO 380
      IF(.NOT.LO$B) GOTO 3376
      WRITE(NF1,3375)
 3375 FORMAT(6X,'COMMON/$TEST$/ NOTEST'/6X,'LOGICAL*1 NOTEST')
      NREL1=NREL1+2
 3376 WRITE(NF1,3377)
      NREL1=NREL1+1
 3377 FORMAT('C$F')
      IF(LO$B) WRITE(NF1,3378)NT
 3378 FORMAT(6X,'IF(NOTEST) GOTO ',5A1)
      IF(LO$B) NREL1=NREL1+1
      IF (.NOT.LOZ) GOTO 371
      WRITE(NF1,375) SEIG,SSTO,EIGA,STOFF,NARG1,NARG1,NRET
  375 FORMAT(       6X,'CALL $NUMBR(',A2,A4,2H,',A2,A4,''',NUMBR$(',I2,')
     1,',I2,',1,I,&',5A1,')')
```

```
      NRF11=NRF11+1
      LCZ=.FALSE.
  371 IF(LCG.AND.NARG1.GT.0)CALL C$P(IRFUN1,IREND ,NARG1,ARGAKT,KA,NRET,
     1EIGA,STOFF,SEIG,SSTO,NF1,NRF11)
      LCG=.FALSE.
      IF(LCF) GOTO 372
  380 IF(K.NE.IRTZ  .OR.  .NOT.LC$B) GOTO 376
      CALL GETREC(A4,K)
      WRITE(NF1,325) NT,(LAA(I),I=6,80)
  325 FORMAT('C$T'/80A1)
      NRF11=NRF11+2
      GOTO 372
  376 IF (K.NE.IRET) GOTO 377
      WRITE (NF1,378) NRET
  378 FORMAT(5A1,' RETURN')
      GO TO 379
  377 CALL GETREC(A4,K)
  374 WRITE(NF1,373) A4
  373 FORMAT(20A4)
  379 NRF11 = NRF11+1
  372 CONTINUE
C 372
CB    ENDE DER AUSGABE VOR DEM NAECHSTEN ENTRY ODER END-STATEMENT
C
  200 NRENT=NRENT+1
      IF (FK.EQ.ENT) GOTO 910
      IF((JEIN+1).NE.JDC) GOTO 910
      JEIN=JEIN+1
      IF (JEIN.LE.JEMAX) GOTO 910
      JEIN=JEMAX
      CALL FEHLER('JEIN.GT.JEMAX',13,8999)
  910 IF (IENT.GE.NRENT) GOTO 30
      IF(.NOT.LCR) NEWMAS=.TRUE.
      IF(LCR.AND.                .NOT.LKKD)
     1CALLFEHLER('FUNCTION DOES NOT EXIST, BUT HAS BEEN ADDED TO MAPLIB'
     2 ,53,843)
   43 IF(LCR.AND.LKKD) WRITE(6  ,44) EIGA,STOFF
   44 FORMAT('OFUNCTION ',A2,A4,' HAS BEEN REPLACED IN MAPLIB')
      IF(.NOT.LCR.AND..NOT.LKKD) WRITE(6  ,45) EIGA,STOFF
   45 FORMAT('OFUNCTION ',A2,A4,' HAS BEEN ADDED TO MAPLIB')
      IF (IENT.EQ.0) GOTO 913
      WRITE(NF1,914) IENT,((NAMENT(I,J),I=1,3),J=1,IENT)
C
CB    AKTIEREN DER ENTRY-NAMEN FUER DIE ERSTELLUNG DES ALIAS-STATEMENTS
CB    MIT FILOUT UND EXEC2  AUF   $$-CARD.
C
C
  914 FORMAT('$$',I2,18A2)
      NRF11=NRF11+1
  913 WRITE(NF1,917)
  917 FORMAT(6X,'END')
      NRF11=NRF11+1
      NRECA(JDC)=NRF110
      NRECE(JDC)=NRF11
      IF(LCS) LARG$(1,JDC)=DOLLAR
      GOTO 10
C     GOTO 10: NAECHSTE FUNCTION
```

```
C
  999 NF1=NF11
C
C 999 LABEL FUER FEHLERAUSGANG MIT ERROR=.TRUE.
C
      KE =KE1
      KA=KA1
      KS=KS1
      IF(ERKFCL) RETURN
      JEIN =JEIN1
      KY=KY1
      ERROR=.TRUE.
      NEWMAS=.FALSE.
 1200 CONTINUE
C
      RETURN
 1000 IF(ERROR) GOTO 999
      GOTO 1200
C
C     GOTO 1200:  NORMALER AUSGANG
C
      END
```

```
      SUBROUTINE KATALC(KAT,I2,KA,KAMAX,ERKKAT,EL,SW,IRFUN,IREND,OUT,
     1NF1,*)
CN    KATALC  -  AUFNAHME VON KOMMENTAREN UND ELEMENTEN IM KATALOG
CD    1.5.70
CB    DAS ELEMENT EL(I2) WIRD, WENN NICHT VORHANDEN,DER LISTE
CB    KAT(I2,KAMAX), DIE BISHER KA ELEMENTE ENTHAELT, HINZUGEFUEGT.
CB    IN DER EINGABE WIRD NACH ZUGEHOERIGEN, MIT DER IDENTIFIKATION SW
CB    VERSEHENEN KOMMENTAREN GESUCHT. (EINGABE-KARTEN IRFUN BIS IREND)
CB    DER NEUE KOMMENTAR WIRD IN ERKKAT GESPEICHERT
CB    FALLS DER KOMMENTAR VON FRUEHEREM ABWEICHT, WIRD WARNUNG AUSGE-
CB    SCHRIEBEN
CB    FALLS ELEMENT BISHER NICHT ERKLAERT WURDE, WIRD WARNUNG AUSGE-
CB    SCHRIEBEN
CB    DIMENSIONSANGABEN MIT VORHERGEHENDEN DOLLAR-ZEICHEN WERDEN AUF
CB    SPALTE 61-72 GERUECKT
CB    FALLS NF1>0 WIRD KOMMENTAR AUF FILE NF1 GESCHRIEBEN
CB    FALLS IREND<0 WERDEN KOMMENTARKARTEN PAARWEISE
CB    ERWARTET
CB    EBENSO FUER  SW='CP',WENN MEHR ALS 1 ELEMENT IN DEN KARTEN VOR-
CB    KOMMT
      COMMON/SKONT4/       IDUMMY(24),
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$0,K$MAX
      COMMON/SDIM1/ LSYN(500),LERL(4000),FIL(80,100)
      LOGICAL*1LSYN,LERL,FIL
      COMMON/SKONT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1   ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,  FUNCDA,NEWMAS,PUNCH
C
      INTEGER*2 KAT(I2,KAMAX),EL(I2),SW,A(40)/40*'  '/,CA/'CP'/
     1,QE(80)/80*'  '/,CP/'$P'/,B/'  '/,BLK2,AE(3)
      INTEGER OUT,ERKKAT(17,KAMAX),A4(17),DAT(2)/'UNKNOWN '/,BLK4
     1 /'    '/,FRAGZ/'  ??'/
      LOGICAL*1 LA(80),GETFIL,LELNEU,LERK,LERNEU,LAEND,LE(2,80),
     1 LBLK/' '/,SWEQCP,NOTERK,LB,LEND,LAE(6),WRTCP
      EQUIVALENCE (LA(1),A(1),A4(1)),(QE(1),LE(1,1))
      EQUIVALENCE(BLK4,BLK2),(B,LB),(LAE(1),AE(1))
C     CATALOGUES EL
      SWEQCP=SW.EQ.CA
CV    SWEQCP=SW.EQ.'CP'   -  ERKLAERUNG DER PARAMETER
      LELNEU=.FALSE.
CV    LELNEU=.TRUE., WENN  EL NOCH NICHT IN KAT VORHANDEN WAR
      LERNEU=.FALSE.
CV    LERNEU=.TRUE., WENN  EL NEU DURCH EINGABE ERKLAERT WIRD
      LAEND=.TRUE.
CV    LAEND=.TRUE.,WENN NEUER KOMMENTAR VON ALTEM ABWEICHT
      WRTCP=.FALSE.
CV    WRTCP=.TRUE.,FALLS KOMMENTAR BEREITS GESCHRIEBEN WURDE
      DO 29 I=1,17
   29 A4(I)=BLK4
      CALL LSTCON(EL,KAT,I2,1,IER,KA,&2)
      GOTO 4
    2 KA=KA+1
      IF(KA.GT.KAMAX) CALL FEHLER('NOT ENOUGH SPACE IN CATALOGUE',29,&99
     19)
      DO 5 J=1,I2
    5 KAT(J,KA)=EL(J)
```

```
      LELNEL=.TRUE.
      IER=KA
      DO 51 J=1,2
   51 ERKKAT(J,KA)=DAT(J)
      DO 52 J=3,17
   52 ERKKAT(J,KA)=BLK4
      IF(SW.EQ.CP .OR. SWEQCP)ERKKAT(14,KA)=FRAGZ
      GOTO 53
    4 LERK=.TRUE.
CV    LERK =.TRUE.,WENN ELEMENT ERKLAERT IST ODER WAR
      DO7J=1,2
      IF(ERKKAT(J,IER).NE.DAT(J)) GOTO 6
    7 CONTINUE
   53 LERK=.FALSE.
    6 IF(IREND.LT.0)GOTO 500
      IR=IRFUN
      M=2
      IF(SWEQCP) M=1
      NOTERK=.NOT.LERK
   21 CALL SUCHR(SW,2,M,IR,IREND,IRSW,&19)
      IF(.NOT.SWEQCP) GOTO 14
      CALL GETREC(A4,IRSW)
      LEND=.FALSE.
      DO 201 I=1,3
  201 AE(I)=A(I+3)
  204 DO 202 I=1,3
      IF(AE(I).NE.EL(I))GOTO 203
  202 CONTINUE
      GOTO 299
  203 IF(LEND)GOTO11
      DO 205 I=2,5
      LB=LAE(I)
      IF(B.EQ.BLK2) GOTO 206
  205 CONTINUE
      GOTO 11
  206 I=I+1
      DO 207 J=I,6
  207 LAE(J)=LBLK
      LEND=.TRUE.
      GOTO 204
  299 IF(NF1.LE.0)GOTO 410
      WRITE (NF1,400) LA
      WRTCP=.TRUE.
      NRF11=NRF11+1
  400 FORMAT(80A1)
  410 LERL(IRSW)=.TRUE.
      IRSW=IRSW+1
      LA(1)=GETFIL(1,IRSW)
      LA(2)=GETFIL(2,IRSW)
      IF(A(1).EQ.CA) GOTO 14
      IF(ERKFOL )    CALL FEHLER('CP-CARDS ARE NOT FOUND IN PAIRS',31,
     1&99)
      GOTO 19
   14 DO 15 I=7,80
      LA(I) = GETFIL(I,IRSW)
   15 LE(1,I)= LA(I)
      LERL(IRSW)=.TRUE.
```

```
      IF(.NOT.SWEQCP.AND.SW.NE.CP) GOTO 300
      CALL SUCHC(QE,'$ ',7,72,I,&300)
      IF(I.EQ.7) GOTO 19
      L=MINO(72,I+12)
      IF(I.GT.60) GO TO 305
      DO 306 J=I,60
  306 LA(J) = LBLK
  305 K=60-I
      I=I+1
      IF(K.EQ.0) GOTO 300
      DO 302 J=I,L
  302 LA(J+K)=LE(1,J)
      I= L+K+1
      IF(I.GT.74) GOTO 300
      DO 307 J= I,74
  307 LA(J)=LBLK
  300 NOTERK=.FALSE.
      DO 309 I=1,34
  309 A(I)=A(I+3)
      LERNEU=.TRUE.
      DO 16 I=1,17
      IF(A4(I).NE.ERKKAT(I,IER)) GOTO 17
   16 CONTINUE
      LAEND=.FALSE.
   17 IF(.NOT.LERK) GOTO 20
      IF(.NOT.LAEND) GOTO 20
      WRITE(6  ,100)(ERKKAT(I,IER),I=1,17),(A4(I),I=1,17),(EL(I),I=
     1 1,12)
  100 FORMAT('0*** WARNING. OLD COMMENT:',17A4/
     1   5X, ' HAS BEEN CHANGED IN:',17A4/ 6X,'FOR THE CATALOGUE-MEMBER:
     1', 3A2)
      FERROR=.TRUE.
   20 DO 18 I=1,17
   18 ERKKAT(I,IER)= A4(I)
   19 IF(NOTERK) WRITE(6  ,416) EL
      IF(NOTERK)FERROR=.TRUE.
  416 FORMAT('0*** WARNING. MAPLIB DOES NOT KNOW   ',3A2)
      IF(NF1.LE.0)GOTO 10
      IF(SWEQCP) GOTO 30
      WRITE(NF1,414)SW,(ERKKAT(I,IER),I=1,17)
  414 FORMAT('C',A2,3X,17A4)
      GOTO 31
   30 IF(WRTOP) GOTO 32
      WRITE(NF1,415)SW,EL
      NRF11=NRF11+1
  415 FORMAT(A2,4X,3A2)
   32 WRITE(NF1,402)SW,(ERKKAT(I,IER),I=1,17)
  402 FORMAT(A2,4X,17A4)
   31 NRF11=NRF11+1
   10 CONTINUE
      RETURN
  500 IRSW=IRFUN
      LA(1)=GETFIL(2,IRSW)
      LA(2)=GETFIL(3,IRSW)
      IF(A(1).NE.SW)CALL FEHLER('COMMENT CARDS ARE NOT FOUND IN PAIRS',
     136,&99)
      GOTO 14
```

```
 11 IR=IRSW+2
    GOTO 21
 99 LERK=.FALSE.
    GOTO 19
999 ERROR =.TRUE.
    KA=KAMAX
    RETURN 1
    END
```

```
      SUBROUTINE  FUNANA(FU,EIGEN,STOFF,NARG,ARG,IR,NAMARG,NF1,NAMIN,*)
CN    FUNANA - ANALYSE VON FUNCTION-,ENTRY- UND SUBROUTINEN-STATEMENTS
CB    FUNANA  STELLT FEST:
CP    EIGEN,STOFF:  RCUTINEN-NAMEN
CP    NARG    - ZAHL DER ARGUMENTE (INCL. DUMMY-ARGUMENT)
CP    ARG     - NAMEN DER ARGUMENTE
CP    NAMARG - PARAMETERLISTE FUER DAS AUSDRUCKEN IN TABWRT (.TAB)
CP    NF1     - AUSGABE-EINHEIT
CP    *       - RUECKSPRUNGLABEL IM FALLE EINES FEHLERS
CB    WENN NARG>=0 WIRD DAS FUNCTION-STATEMENT ODER ENTRY-STATEMENT
CB    VERSEHEN MIT SLASHS(SCHRAEGSTRICHEN) AUF DER EINHEIT NF1 AUSGE-
CB    GEBEN
CB    WENN DIE ARBUMENTENLISTE MEHR ALS 24 ZEICHEN ENTHAELT, WIRD SIE
CB    HIER VERKUERZT IN NAMARG ABGESPEICHERT.
C
      COMMON/SDIM1/ LSYN(500),LERL(4000),FIL(8000)
      LOGICAL*1 LSYN,LERL,FIL
      COMMON/SKONT1/ ERROR,FEHLOR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1   ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FEHLOR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,  FUNCDA,NEWMAS,PUNCH
      COMMON/SKONT4/       IDUMMY(24),
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$0,K$MAX
      INTEGER OUT,PRT
      INTEGER FU(2),FUN/'FUNC'/,SUBR/'SUBR'/
      INTEGER*2 Q(396),BLK2/'  '/,C/'C '/,NARG,SL/'/ '/,KOM/', '/
     1,KOLON/'; '/
      LOGICAL*1 LQ(2,396),GETFIL,EIGEN(2),STOFF(4),ARG(6,48),LBLK/' '/,
     1 ENDE,LCC,NAMARG(24),LSYS
      DATA MAXC/396/
      EQUIVALENCE (Q(1),LQ(1,1))
      CALL FSPIE
      LSYS=NARG.LT.0
      LCC =.FALSE.
      DO 1 I=1,MAXC
    1 Q(I) = BLK2
C
CB    UMSPEICHERUNG DES FUNCTION-STATEMENTS IN  Q
      JR=IR+1
      K=IR
      DO 2 I=1,6
    5 LQ(1,1)= GETFIL(1,JR)
      IF(Q(1).NE.C) GOTO 4
      LCC=.TRUE.
      JR=JR+1
      GOTO 5
    4 LQ(1,1) = GETFIL(6,JR)
      IF(Q(1).EQ.BLK2)GOTO 3
      IF(LCC)CALL FEHLER('COMMENT-CARD BEFORE THE LAST CARD OF THE ENTRY
     1- OR FUNCTION-STATEMENT',70,&9)
      K=K+1
    2 JR=JR+1
      CALL FEHLER('THE FUNCTION- OR ENTRY-STATEMENT CONSTITS OF MORE THA
     1N SIX CARDS',64,&9)
    9 MAXC=396
      RETURN1
    3 JR=K
```

```
      I=1
      DO 6 KR=IR,JR
      LERL(KR) = .TRUE.
      DO 7 II= 7,72
      LQ(1,I)= GETFIL(II,KR)
    7 I=I+1
    6 CONTINUE
      MC=5
      IF(FU(1).EQ.FUN) MC=8
      IF(FU(1).EQ.SUBR)MC=10
      MMC=MC
      MAXC=I-1
C
CB    ELIMINATION VON BLANKS UND SLASHS
C
      K=MAXC-1
      DO 24 J =9,MAXC
      DO 20 I= MMC,K
      IF(Q(I).EQ.BLK2 .OR. Q(I).EQ.SL) GOTO 21
   20 CONTINUE
      GOTO 23
   21 MMC= I
      DO 22 KK= I,K
   22 Q(KK)=Q(KK+1)
      Q(MAXC)=BLK2
   24 CONTINUE
      CALL SUCHC(Q,BLK2,MC,MAXC,I,&23)
      MAXC=I
C
CB    ANALYSE
C
   23 CALL SUCHC(Q,'( ',MC,MAXC,KLAUFO,&25)
      IF((KLAUFO-MC).GT.7)CALL FEHLER('THE FUNCTION-NAME CONSISTS OF MOR
     1E THAN SIX CHARACTERS',54,& 9)
   11 EIGEN(1)=LQ(1,MC+1)
      EIGEN(2)=LQ(1,MC+2)
      DO 26 I=1,4
   26 STOFF(I)=LBLK
      ENDE = .FALSE.
      IF(LSYS) GOTO 400
      DO 31 I=1,288
   31 ARG(I,1)=LBLK
      DO 12 I=1,24
   12 NAMARG(I)=LBLK
  400 I=MC+3
      J=KLAUFO-1
      IF(LSYS.AND.J.LT.I)GOTO 99
      KLAUF=KLAUFO
      L=1
      DO 27 K=I,J
      STOFF(L)=LQ(1,K)
   27 L=L+1
      IF(LSYS) GOTO 99
      CALL SUCHC(Q,') ',KLAUF,MAXC,KLZU,&28)
      NA= 1
C     ANALYSE DER ARGUMENTEN-LISTE
      KOMMA = KLAUF+1
```

```
   33 CALL SUCHC(Q,KOM,KOMMA,KLZU,KOMMA,&29)
   34 K=KOMMA-KLAUF-1
      IF(K.GT.6) CALL FEHLER('THE ARGUMENT-NAME CONSISTS OF MORE THAN SI
     1X CHARACTERS',54,&9)
      IF(K.EQ.0) CALL FEHLER('ERROR IN ARGUMENT-LIST',22,&9)
      DO 32 I=1,K
   32 ARG(I,NA)=LQ(1,KLAUF+I)
      IF(ENDE) GOTO 199
      KLAUF=KOMMA
      KOMMA=KOMMA+1
      NA=NA+1
      IF(NA.GT.48) CALL FEHLER('MORE THAN 48 PARAMETERS',23,&9)
      GOTO 33
   29 ENDE=.TRUE.
      KOMMA=KLZU
      GOTO 34
C
C&    AUSGABE AUF NF1
C
  199 IF (NA.EQ.1) WRITE(NF1,300) FU,BLK2,EIGEN,STOFF,Q(KLAUF0),SL,
     1                            (ARG(I,1),I=1,6),SL,Q(KLZU)
      IF (NA.EQ.1) GO TO 200
      WRITE(NF1,300) FU,BLK2,EIGEN,STOFF,Q(KLAUF0),SL,(ARG(I,1),I=1,6),
     1               (SL,KOM,SL,(ARG(I,J),I=1,6),J=2,NA),SL,Q(KLZU)
  300 FORMAT(6X,2A4,53A1/(5X,'1',63A1))
  200 NRF11=NRF11+(NA+9)/7
      IF (FU(1) .NE. FUN) GOTO 99
C
C&    ABSPEICHERUNG DER PARAMETER-LISTE IN NAMARG
C
      IF((KLZU-KLAUF0).GT.23)CALL FEHLER('THE TOTAL ARGUMENT-LIST CANNOT
     1 BE STORED IN 24 BYTES; IT MUST BE ABBREVIATED',76,& 201)
  204 J=1
      KOMMA=KLAUF0
      DO 209 I=1,NAMIN
      CALL SUCHC(Q,KOM,KOMMA,KLZU,KOMMA,&210)
      KOMMA=KOMMA+1
  209 CONTINUE
      Q(KOMMA-1) =KOLON
  210 CONTINUE
      DO 205 I=KLAUF0,KLZU
      NAMARG(J)=LQ(1,I)
  205 J=J+1
      NARG=NA
   99 RETURN
C
C&    VERKUERZUNG DER PARAMETER-LISTE
C
  201 J=KLAUF0+2
      KLAUF=KLAUF0
  206 CALL SUCHC(Q,KOM ,KLAUF,KLZU,KOMMA,&203)
      KLAUF=KOMMA+1
      Q(J)=Q(KOMMA)
      Q(J+1)=Q(KOMMA+1)
      J=J+2
      GOTO 206
  203 KLZU=J
```

```
      G(J)=G(KLZU)
      IF((KLZU-KLAUFO).GT.23)CALL FEHLER('THE ABBREVIATED ARGUMENT-LIST
     1CANNOT BE STORED TOO',50,&208)
  208 KLZU=MIN0(J,(KLAUFO+23))
      GOTO 204
C
CB    FEHLER-UND SONDERFAELLE
C
   25 IF(LSYS) GOTO 401
      CALL FEHLER('THE ARGUMENT-LIST DOES NOT BEGIN WITH A BRACKET',47,
     189)
  401 KLAUFO=MC+7
      GOTO 11
   28 CALL FEHLER('THE ARGUMENT-LIST DOES NOT END WITH A BRACKET',45,89)
      END
```

```
      SUBROUTINE NRRETU(IR1,IR2,IR ,N,*)
CN    NRRETU - NUMBER OF THE RETURN-STATEMENT
CB    FUER DIE FUNCTION ZWISCHEN RECORD IR1 UND IR2 SUCHT NRRETU NACH
CB    DEM ERSTEN RETURN-STATEMENT MIT EINER STATEMENT-LABEL. WENN KEINS
CB    VORHANDEN IST, WIRD DAS LETZTE RETURN-STATEMENT UND EIN NOCH NICHT
CB    VERWENDETES LABEL ERMITTELT
CP    IR1,IR2  BEREICH
CP    IR     RECORD-NR. DES RETURN-STATEMENTS
CP    N      LABEL
CP    * ,    RUECKSPRUNGLABEL FUER FEHLERAUSGANG
      LOGICAL*1 LQ(2,5),GETFIL,FIRST,N(5)
      INTEGER*2 BLK2/'  '/,C/'C '/
     1   ,Q(5)/5*'  '/
      INTEGER*2 NR(5),IN(5)
      EQUIVALENCE(LQ(1,1),Q(1))
      I1=IR1
      FIRST=.FALSE.
      CALL NEWNR(IN,NR)
    2 CALL SUCHR('RETURN',6,7,I1,IR2,IR,&10)
      FIRST=.TRUE.
      DO 30 I=1,5
   30 LQ(1,I)=GETFIL(I,IR)
      IF(     Q(1).EQ.C) GOTO 345
      DO 1 I=1,5
      IF(Q(I).NE.BLK2) GOTO 5
    1 CONTINUE
  345 I1=IR+1
      GOTO 2
   10 IF(.NOT.FIRST) CALL FEHLER('RETURN-STATEMENT NOT FOUND',26,&99)
    7 CALL PERNBR(IR1,IR2,NR,&6)
      DO 50 I=1,5
   50 Q(I)=NR(I)
      GOTO 12
    6 CALL STATNR(IN,NR,&31)
      GOTO 7
   31 CALL FEHLER('ALL STATEMENT-NBRS. ARE USED, BUT NO IN A RETURN-STAT
     1EMENT',58,&99)
   99 RETURN 1
    5 IF(Q(5).NE.BLK2) GOTO 12
      DO 43 I=1,4
   43 Q(6-I)=Q(5-I)
      Q(1)=BLK2
      GO TO 5
   12 DO 13 I=1,5
   13 N(I)=LQ(1,I)
  100 RETURN
      END
```

```
      SUBROUTINE NRTEST(NT,N,IR1,IR2,IT,*)
CN    NRTEST -STATEMENT-NUMBER DES CONTINUE-STATEMENTS NACH C$T
CP    NT     STATEMENT-NR. ALS ZEICHENKETTE
CP    N      STATEMENT-NR. DES RETURN-STATEMENTS, ALS ZEICHENKETTE
CP    IR1    RECORD-NUMBER DES BLOCK-ANFANGS
CP    IR2    RECORD-NUMBER DES BLOCK-ENDES
CP    *      RUECKSPRUNGLABEL, FUER DEN FALL, DAS KEINS DER 99999 ST.NRS.
CP    FREI IST
      LOGICAL*1 NT(5),N(5),LNR(2,5)
     1,GETFIL
      INTEGER*2 IN(5),NE(5),NR(5)/5*'  '/
     1,C/'C '/,BLK2/'  '/
      EQUIVALENCE (LNR(1,1),NR(1))
   13 DO 10 I=1,5
      LNR(1,I)=GETFIL(I,IT)
      IF(NR(I).EQ.C) GOTO 11
      IF(NR(I).NE.BLK2) GOTO 12
   10 CONTINUE
      DO 1 I=1,5
      LNR(1,I)=N(I)
    1 NE(I)=NR(I)
      CALL NEWNR(IN,NR)
    5 DO 2  I=1,5
      IF(NE(I).NE.NR(I)) GOTO 6
    2 CONTINUE
      GOTO 3
    6 CALL PERNBR(IR1,IR2,NR,&3)
      DO 4 I=1,5
    4 NT(I)=LNR(1,I)
      RETURN
    3 CALL STATNR(IN,NR,&31)
      GOTO 5
   31 CALL FEHLER('ALL STATEMENT-NUMBERS ARE USED *NRTEST',38,&7)
    7 RETURN 1
   11 IT=IT+1
      GOTO 13
   12 DO 14 I=1,5
   14 NT(I)=GETFIL(I,IT)
      RETURN
      END
```

```
      SUBROUTINE  PERNBR(IR1,IR2,NR,*)
CN    PERNBR   UEBERPRUEFUNG, OB DIE STATEMENT-NR NR NICHT SCHON VER-
CB             WENDET WURDE.
CB    WENN NEIN: RETURN
CB    WENN JA  : RETURN 1
CP    IR1    ANFANGS-RECORD DER FUNKTION
CP    IR2    END     -RECORD DER FUNKTION
CP    NR     ZEICHENKETTE MIT BLANKS
      INTEGER*2 NR(5),Q(5)/5*'  '/,CO/'C '/,BLK2/'  '/
      LOGICAL*1 LQ(2,5),GETFIL
      EQUIVALENCE(LQ(1,1),Q(1))
      DO 3 IRT=IR1,IR2
      DO 4 II=1,5
    4 LQ(1,II)=GETFIL(II,IRT)
      IF(Q(1).EQ.CO)GOTO 3
      IF(Q(1).NE.BLK2.AND.Q(1).NE.NR(1)) GOTO 3
      DO 15 II=2,5
      IF(Q(II).NE.NR(II)) GOTO 3
   15 CONTINUE
      RETURN 1
    3 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE STATNR(IN,NR,*)
CN    STATNR -      STATNR=STATNR-1
CB    STATNR LIEFERT DIE UM EINS ERNIEDRIGTE NR.
CP    IN(5)    DIE 5 ZIFFERN DER STATMENT-NR. ALS ZAHLENWERT
CP    NR(5)    DIE 5 ZIFFERN DER STATMENT-NR. ALS ZEICHEN
CP    *        RUECKSPRUNGLABEL FUER DEN FALL, DASS DIE ALTE NR 00001 WAR
      INTEGER*2 ZAHL(10)/'0 1 2 3 4 5 6 7 8 9 '/,IN(5),NR(5)
      I=5
   32 IN(I)=IN(I)-1
      NR(I)=ZAHL(IN(I))
      IF(IN(I).NE.0) GOTO 7
      IN(I)=10
      NR(I)=ZAHL(10)
      I=I-1
      IF(I) 31,31,32
   31 RETURN 1
      ENTRY NEWNR(IN,NR)
CN    NEWNR
CB    INITIALISIERUNG VON IN UND NR MIT 99999
      DO 30 I=1,5
      NR(I)=ZAHL(10)
   30 IN(I)=10
    7 RETURN
      END
```

```
      SUBROUTINE  C$P (IR1,IR2,NARG,ARGAKT,KA,NRET,EIG,STOFF,SEIG,SSTO,
     1NF1,NRF11)
CB    EINEAU DER SUBROUTINE $RANGE ZUR UEBERPRUEFUNG DES GUELTIGKEITS-
CB    BEREICHS DER PARAMETER
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTOA(100),NRSTOE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON/SDIM2/ IDUMMY(560),KATARG(3,50),IDUMM(150)
      INTEGER*2 IDUMM,KATARG
      LOGICAL*1 ARGAKT(6,48),LB,LE(2,65),LA(90),LERK(68,1),LKOM/','/,
     1UND/'&'/,NRET(5),GETFIL,LAPO/1H'/,KLZU/')'/
      INTEGER*2 EIG,SEIG,B/'  '/,BLK2/'  '/,Q (65)/65*'  '/,GLEICH/'= '/
     1,KLEIN/'< '/
     2,NARG
      INTEGER STOFF,SSTO
      EQUIVALENCE(B,LB),(LE(1,1),Q(1)),(ERKARG(1,1),LERK(1,1))
      DO 1000 IN =1,NARG
      DO 1 IT = 2,6
      LB=ARGAKT(IT,IN)
      IF(B.EQ.BLK2) GOTO 2
    1 CONTINUE
      IT=7
    2 I=MINO(IT,6)
      CALL SUCHR(ARGAKT(1,IN),I,7,IR1,IR2,K,&1000)
      IT=IT+6
      DO 3 J=IT,72
    3 LE(1,J-7)=GETFIL(J,K)
      IKOMMA = 0
      IZ = 1
      IT = IT-7
      DO 11 I= 1,65
      IF(Q(I).EQ.BLK2.OR.Q(I).EQ.GLEICH) GOTO 11
      IF(Q(I).NE.KLEIN) GOTO 12
      IKOMMA= IKOMMA+1
      LA(IZ)=LKOM
      GOTO 13
   12 LA(IZ)=LE(1,I)
   13 IZ=IZ+1
   11 CONTINUE
      IF(IKOMMA.NE.4) GOTO 1000
      LA(IZ)= LKOM
      LA(IZ+1)=LAPO
      CALL LSTCON(ARGAKT(1,IN),KATARG,3,1,J,KA,&1000)
      IZ=IZ+2
      DO 14 K=55,66
      LA(IZ)=LERK(K,J)
   14 IZ=IZ+1
      LA(IZ)=LAPO
      LA(IZ+1)=LKOM
      LA(IZ+2)=UND
      IZ=IZ+3
      DO 15 K=1,5
      LA(IZ)=NRET(K)
   15 IZ=IZ+1
      LA(IZ)=KLZU
```

```
      WRITE(NF1,100) SEIG,SSTO,EIG,STOFF,(ARGAKT(I,IN),I=1,6),(LA(I),I=
     1 1,IZ)
  100 FORMAT(6X,'CALL $RANGE(',A2,A4,2H,',A2,A4,3H',',6A1,2H',,29A1/
     15X,'1',66A1)
      NRF11=NRF11+1
      IF(IZ.GE.29) NRF11=NRF11+1
 1000 CONTINUE
      RETURN
      END
```

```
      FUNCTION NUMERI(L1,L2)
CN    NUMERI
CB    BESTIMMUNG DES ZAHLENWERTES EINES IN DER FORM VON ZWEI ZIFFERN
CB    ANGEGEBENEN INTEGERS
      LOGICAL*1 L1,L2,LE1,LE2
      INTEGER*2 I1/'  '/,I2/'  '/,ZAHL(11)/'0 1 2 3 4 5 6 7 8 9   '/
      EQUIVALENCE(I1,LE1),(I2,LE2)
      LE1=L1
      LE2=L2
      IF(I1.EQ.ZAHL(11)) GOTO 10
      DO 1 I=1,10
      IF(I1.EQ.ZAHL(I)) GOTO 2
    1 CONTINUE
    5 J=0
      CALL FEHLER('NO NUMBER',5,&999)
   10 I=1
    2 J=I-1
      IF(I2.EQ.ZAHL(11)) GOTO 999
      J=J*10
      DO 3 I=1,10
      IF(I1.EQ.ZAHL(I)) GOTO 4
    3 CONTINUE
      GOTO 5
    4 J=J+I-1
  999 NUMERI=J
      RETURN
      END
```

```
      SUBROUTINE ADDSYS
CN    ADDSYS
CB    DIESE ROUTINE REGISTRIERT SYSTEM-ROUTINEN.
CB    VERGL. ADDFUN
      COMMON/SDIM4/ STO( 500),ARG(6,500),NREGA(500),NREGE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTGA(100),NRSTGE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON /SDIM2/IDUMMY (635)  ,SYSTEM(3,50)
      INTEGER*2 SYSTEM
      COMMON/SDIM1/ LSYN(500),LERL(4000),FIL(80,100)
      LOGICAL*1LSYN,LERL,FIL
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      COMMON/SKONT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1   ERKFOL,FUNODA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,  FUNODA,NEWMAS,PUNCH
      INTEGER*2 NARG/-1/,NAM2,NAM42(2)
      LOGICAL*1 LDO,LOR,ARGAKT(6),ALIAS(6,6),LBLK/' '/,L(80),GETFIL
     1,LORPRO
      INTEGER EEND/'END '/
     1,A4(20)
      REAL*8 FK,FUN(3)/'FUNCTION','SUBROUTI','ENTRY   '/
      INTEGER OHAR(9)/8,8,5/
      EQUIVALENCE(NAM4,NAM42(1)),(A4(1),L(1),ARGAKT(1),LEND)
      CALL FSPIE
      CALL FILFIL(INP,NDIMAX,FIL,LERL)
      FUNODA=.FALSE.
      IRMAX=IAV-2
      IRANF=1
      LORPRO=.FALSE.
      CALL SUCHR('O$R',3,1,1,1,I,&1000)
      LORPRO=.TRUE.
      IRANF=2
 1000 DO 1310 I=1,3
 1310 L(I)=GETFIL(I,IRANF)
      IF(LEND.EQ.EEND) GOTO 500
      CALL FROMRO(IRANF)
      NRFO=NRF11+1
      LDO=.FALSE.
      I=IRANF
    2 CALL SUCHR('END',3,7,I,IRMAX,IREND,&999)
      CALL SUCHR('O$O',3,1,IREND-1,IREND-1,I,&1)
      I=IREND+1
      GOTO 2
    1 CALL FILOUT(NDA,IRANF,IREND,'SYSTEM  ROUTINES  INPUT',23)
      IREND=IREND-1
      CALL SUCHR(FUN(1),8,7,IRANF,IRANF,I,&3)
      FK=FUN(1)
      GOTO 4
    3 CALL SUCHR(FUN(2),8,7,IRANF,IRANF,I,&998)
      FK=FUN(2)
    4 CALL FUNANA(FK,NAM2,NAM4,NARG,ARGAKT,IRANF,ARGAKT,NF1,I,&996)
```

```
      K$=K$+1
      IF(K$.GT.K$MAX) CALL FEHLER('K$.GT.K$MAX *ADDSYS',19,&990)
      SYSTEM(1,K$)=NAM2
      SYSTEM(2,K$)=NAM42(1)
      SYSTEM(3,K$)=NAM42(2)
   23 LCR=LCRPRC
      CALL LSTCON(SYSTEM(1,K$),SYSTEM,3,1,K$D,K$-1,&20)
      K$=K$-1
      LDC=.TRUE.
      IF(LCR) GOTO 21
      CALL SUCHR('C$R',3,1,IRANF,IREND,I,&21)
      LCR=.TRUE.
      GOTO 21
   20 K$D=K$
   21 IF(LDC.AND..NOT.LCR)CALL FEHLER('DOUBLE INPUT, NO REPLACE-OPTION',
     131,&&94)
      IE=0
      DO 32 I=1,36
   32 ALIAS(1,1)=LBLK
      DO 30 I=1,3
      J=IRANF+1
    6 CALL SUCHR(FUN(I),CHAR(I),7,J,IREND,J,&30)
      CALL SUCHR('C',1,1,J,J,K,&33)
      GOTO 34
   33 CONTINUE
      IF(IE.GE.6) CALL FEHLER('THERE ARE MORE THAN SIX ALIAS, ONLY THE F
     1IRST SIX ARE CATALOGUED',64,&31)
      IE=IE+1
      CALL ADDALI(I,ALIAS(1,IE),J)
   34 J=J+1
      GOTO 6
   30 CONTINUE
   31 DO 11 I=IRANF,IREND
      CALL GETREC(A4,I)
      WRITE(NF1,100) A4
  100 FORMAT(20A4)
      NRF11=NRF11+1
   11 CONTINUE
      WRITE(NF1,101)IE,ALIAS
  101 FORMAT('$$',I2,36A1/6X,'END')
      NRF11=NRF11+2
      IF(LDC.AND.LCR) WRITE(6  ,102) NAM2,NAM42
  102 FORMAT('0THE SYSTEM MEMBER *',3A2,'* HAS BEEN REPLACED')
      IF(.NOT.LDC) WRITE(6  ,103) NAM2,NAM42
  103 FORMAT('0THE SYSTEM MEMBER *',3A2,'* HAS BEEN ADDED TO THE SYSTEM'
     1)
      NRECSA(K$D)=NRFC
      NRECSE(K$D)=NRF11
      IRANF=IREND+2
      IF(IRANF.LT.IRMAX) GOTO 1000
  500 RETURN
  999 CALL FEHLER('END-STATEMENT NOT FOUND',23,&899)
  899 CALL FILCUT(NDA,IRANF,IRMAX,'NOT ADDED SYSTEM ROUTINES INPUT',31)
      GOTO 500
  998 CALL FEHLER('THE FIRST CARD OF THIS BLOCK IS NEITHER A SUBROUTINE-
     1 NOR A FUNCTION-STATEMENT',78,&894)
  894 IRANF=IREND+2
```

```
      IF(IRANF.GE.IRMAX) GOTO 500
      GOTO 1000
  896 CALL FEHLER('ERROR IN THE FIRST STATEMENT',28,&894)
  890 ERROR=.TRUE.
      GOTO 899
      END
```

```
      SUBROUTINE ADDALI(I,ALIAS,J)
CN    ADDALI
CB    REGISTRIERUNG VON ALIAS-NAMEN IN ADDSYS
      LOGICAL*1 ALIAS(6),LB,LBLK/' '/,GETFIL
      INTEGER*2 BLK2/'  '/,B,KLAUF/'( '/
      EQUIVALENCE (B,LB)
      B=BLK2
      DO 10 K=1,6
   10 ALIAS(K)=LBLK
      I1=14
      IF(I-2) 11,12,13
   12 I1=16
      GOTO 11
   13 I1=12
   11 IZ=1
      DO 1 K=I1,30
      LB=GETFIL(K,J)
      IF(B.EQ.BLK2)GOTO1
      IF(B.EQ.KLAUF)GOTO 2
      ALIAS(IZ)=LB
      IZ=IZ+1
      IF(IZ.GT.6) GOTO 2
    1 CONTINUE
    2 RETURN
      END
```

```
      SUBROUTINE PSCAT(S,P)
C
CN    PSCAT  - ERSTELLUNG DER MASTER-ROUTINEN
CB    PSCAT SCHREIBT FUER DIE KATALOGISIERTEN FUNCTIONEN ALLE
CB    MASTER-FUNCTIONEN AUF DIE EINHEIT P IN FORTRAN
CP    P      AUSGABEEINHEIT  (P=NF1)
CP    $      KENNZEICHEN DER MASTERFUNCTIONS
C
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTCA(100),NRSTCE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON/SDIM2/ EIG(500),NARG(500),KATEIG(100),SYNEIG(20),KATARG
     1(3,50),SYSTEM(3,50)
      INTEGER*2 EIG,NARG,KATEIG,SYNEIG,KATARG,SYSTEM
      COMMON/SKONT8/RELDAT,RELUHR,DAT,UHR
      REAL*8 RELDAT,RELUHR,DAT,UHR
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      COMMON/SKONT2/ NAMAX,NAMAXO
      INTEGER*2 NAMAX,NAMAXO
      COMMON/SDIM1/ LSYN(500),LERL(4000),FIL(80,100)
      LOGICAL*1LSYN,LERL,FIL
      INTEGER*2 EIGKAT(100),NEIG(100)
      INTEGER STOKAT(100)
      EQUIVALENCE(FIL(1,1),STOKAT(1)),(FIL(1,6),EIGKAT(1)),(FIL(41,8),
     1 NEIG(1))
C
C
      INTEGER P
      LOGICAL*1 $,TRS(4)/'/',/A'/,TRA(3)/''',''',TRK(2)/',A'/,KZU/')'/,
     1          SL/'/'/,AP/''''/
C
      CALL FINDII(NRF11,P)
      CALL DELCOM
      N$$A=NRF11+1
      NARG=NAMAX
C
C     OBERSTE EBENE   FUNCTION  $$$$$$
C
      WRITE(P,1000) ($,I=1,6),((TRS(I),I=1,4),J,J=1,NARG),SL,KZU
C     FUNCTION $$$$$$(/EIG/,/STOFF/,/A1/,/A2/,/A3/)
 1000 FORMAT(6X,'FUNCTION ' ,6A1,'(/EIG/,/STOFF',7(4A1,I1),
     1       4(3A1/5X,'1',A1,I2,10(4A1,I2)))
      WRITE(P,1050) DAT
C
CN    ***  M A P L I B  ***    RETRIEVAL  FUNCTION
C
CA    MAPLIB-UTILITY
CD    07.07.70
 1050 FORMAT('C'/'CN    ***  M A P L I B  ***    RETRIEVAL  FUNCTION'/
     1 'C'/'CA    MAPLIB-UTILITY'/'CD',4X,A8)
      IF(KS.GT.1)GOTO5001
      WRITE(P,1100)KS,KATSTO(1),AP,SL

                                A7- 44
```

```
      WRITE(P,1200)MARG,KS,KZU
      GOTO 5002
 5001 CONTINUE
      WRITE(P,1100) KS,KATSTO(1),((TRA(I),I=1,3),KATSTO(J),J=2,KS),AP,SL
C     COMMON /$NUMB$/ J,LJ
C     LOGICAL*1 LJ
C     INTEGER*2 EIG
C     INTEGER STOFF,STOKAT( 13)/'4961','4981','4988','HEV ','B4C ',
C    1'UC  ','PUC ','UPUC','NA  ','NAV ','NAL ','NALS','NAVS'/
 1100 FORMAT(6X,'COMMON /$NUMB$/ J,LJ'/6X,'LOGICAL*1 LJ'
     1       /6X,'INTEGER*2 EIG'/6X,'INTEGER STOFF,STOKAT(',I3,')/''',
     2        A4,4(3A1,A4),11(2A1/5X,'1',A1,A4,8(3A1,A4)))
      WRITE (P,1200) MARG,KS,(TRA(2),I,I=2,KS),KZU
C     J=NUMBR$( 3)-2
C     IF (J.LE.-1) GOTO 9996
C     LJ=.FALSE.
C     DO 9998 I=1, 13
C     IF (STOFF.EQ.STOKAT(I)) GOTO ( 1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,
C    112,13)
 5002 CONTINUE
 1200 FORMAT(6X,'J=NUMBR$(',I2,')-2'/6X,'IF (J.LE.-1) GOTO 9996'
     1       /6X,'LJ=.FALSE.'/6X,'DO 9998 I=1,'I3
     2       /6X,'IF (STOFF.EQ.STOKAT(I)) GOTO ( 1',10(A1,I2),
     3        4(A1/5X,'1',I2,20(A1,I2)),A1/5X,'1',I2,4(A1,I3))
      WRITE (P,1300) ($,I=1,6)
C    1 ,I
C9998 CONTINUE
C9996 $$$$$$=ERR$$$(STOFF)
C9999 RETURN
 1300 FORMAT(5X,'1 ,I'/' 9998 CONTINUE'
     1       /' 9996 ',6A1,'=ERR$$$(STOFF)'/' 9999 RETURN')
      NRF11=(MARG+3)/11+(KS+4)/9+(KS+10)/21+2*KS+20 + NRF11
      DO 1 I=1,KS
      M=0
      DO 101 J=1,JEIN
      IF(KATSTO(I).NE.STO(J)) GOTO 101
      IF(NARG(J).GT.M) M=NARG(J)
  101 CONTINUE
      WRITE (P,1400) I,($,J=1,8),KATSTO(I),((TRK(J),J=1,2),K,K=1,M),KZU
C   3 $$$$$$=$$4988(EIG,A 1)
      NRF11=NRF11+(M+4)/16
 1400 FORMAT(2X,I3,' ',6A1,'=',2A1,A4,'(EIG',12(2A1,I2),
     1       3(/5X,'1',16(2A1,I2)))
    1 WRITE (P,1500)
 1500 FORMAT (6X,'GO TO 9999')
C     GO TO 9999
      WRITE (P,1600)
C     END
```

```
 1600 FORMAT (6X,'END')
      N$$E=NRF11
C
C       MASTER-PROGRAMME DER MITTLEREN EBENE
C
C      STOFF-PROGRAMME
C      $$****
C
       DO11 I=1,KS
       NRSTOA(I)=NRF11+1
       M=0
       L=0
       DO 2 J=1,JEIN
       IF (KATSTO(I) .NE. STO(J)) GOTO 2
       L=L+1
       EIGKAT(L)=EIG(J)
       NEIG(L)=J
       IF(NARG(J).GT.M)M=NARG(J)
     2 CONTINUE
       IF(L.EQ.0) GOTO 11
       WRITE (P,2000) $,$,KATSTO(I),((TRS(J),J=1,4),K,K=1,M),SL,KZU
C      FUNCTION $$4988(/EIG/,/A1/)
 2000 FORMAT(6X,'FUNCTION ' ,2A1,A4,'(/EIG',8(4A1,I1),
     1        4(3A1/5X,'1',A1,I2,10(4A1,I2)))
       WRITE(P,1050) DAT
C
CN     ***  M A P L I B  ***     RETRIEVAL  FUNCTION
C
CA     MAPLIB-UTILITY
CD     07.07.70
       WRITE(P,1051)(ERKSTO(J,I),J=1,17)
C$M    STAHL 4988
 1051 FORMAT('C$M',3X,17A4)
       IF (L.GT.1) GOTO 6
       WRITE(P,2100) KATSTO(I),L,EIGKAT(1),AP,SL
       WRITE(P,2200) M,L,KZU
       GOTO 2201
     6 WRITE(P,2100)KATSTO(I),L,EIGKAT(1),((TRA(J),J=1,3),EIGKAT(K),
      1              K=2,L),AP,SL
C      COMMON /$NUMB$/ J,LJ
C      LOGICAL*1 LJ
C      INTEGER*2 EIG,VALUE(4)/'4988    '/,EIGKAT(  6)/'FT','RO','GA',
C     1'EM','CP','WL'/
 2100 FORMAT(6X,'COMMON /$NUMB$/ J,LJ'/6X,'LOGICAL*1 LJ'
     1       /6X,'INTEGER*2 EIG,VALUE(4)/''',A4,'    ''/,EIGKAT(',I3,
     2        ')/''',A2,2(3A1,A2),7(2A1/5X,'1',A1,A2,I2(3A1,A2)))
     7 WRITE (P,2200) M,L,(TRA(2),J,J=2,L),KZU
C      IF (LJ) J=NUMBR$( 1)-1
C      LJ=.FALSE.
C      DO 9998 I=1,  6
C      IF (EIG.EQ.EIGKAT(I)) GOTO(1, 2, 3, 4, 5, 6)
 2200 FORMAT(6X,'IF (LJ) J=NUMBR$(',I2,')-1'/6X,'LJ=.FALSE.'
     1       /6X,'DO 9998 I=1,',I3/6X,'IF (EIG.EQ.EIGKAT(I)) GOTO(1',
     2        11(A1,I2),4(A1/5X,'1',I2,20(A1,I2)),A1/5X,'1',I2,3(A1,I3))
 2201 CONTINUE
       WRITE (P,2300) $,$,KATSTO(I)
C     1 ,I
```

```
C9998 CONTINUE
C     VALUE(3)=EIG
C     $$4988=ERR$X$(VALUE)
C9999 LJ=.TRUE.
C     RETURN
 2300 FORMAT(    5X,'1 ,I'/' 9998 CONTINUE'/6X,'VALUE(3)=EIG'
     1      /6X,2A1,A4,'=ERR$X$(VALUE)'/' 9999 LJ=.TRUE.'/6X,'RETURN')
      NRF11=21+(M+2)/11+(L+11)/14+(L+9)/21+2*L+NRF11
      DO 30 J=1,L
      K=NEIG(J)
      STOKAT(J)=KATSTO(I)
      IF(.NOT.LSYN(K)) GOTO 30
      KK=NRSYN(K)
      EIGKAT(J)=SYNEIG(KK)
      STOKAT(J)=SYNSTO(KK)
   30 CONTINUE
      DO 3 J=1,L
      NN=NARG(NEIG(J))
      IF (NN .GT. 1)  GOTO 5
      WRITE(P,2400) J,$,$,KATSTO(I),EIGKAT(J),STOKAT(J)
C    4 $$4988=EM4988(A1)
 2400 FORMAT(2X,I3,' ',2A1,A4,'=',A2,A4,'(A1)')
      GOTO 4
    5 WRITE(P,2500) J,$,$,KATSTO(I),EIGKAT(J),STOKAT(J),
     1                ((TRK(K),K=1,2),N,N=2,NN),KZU
 2500 FORMAT(2X,I3,' ',2A1,A4,'=',A2,A4,'(A1',12(2A1,I2),
     1       3(/5X,'1',16(2A1,I2)))
    4 WRITE (P,1500)
C     GO TO 9999
      NRF11=NRF11+(NN+3)/16
    3 CONTINUE
      WRITE(P,1600)
C     END
```

```
      NRSTCE(I)=NRF11
   11 CONTINUE
C
C     EIGENSCHAFTS-PROGRAMME
C     **$$$$
C
      DO 12 I=1,KE
      NREIGA(I)=NRF11+1
      M=0
      L=0
      DO 13 J=1,JEIN
      IF (KATEIG(I) .NE. EIG(J)) GOTO 13
      L=L+1
      STOKAT(L)=STO(J)
      NEIG(L)=J
      IF(NARG(J).GT.M)M=NARG(J)
   13 CONTINUE
      IF(L.EQ.0) GOTO 12
      WRITE (P,3000) KATEIG(I),$,$,$,$,((TRS(J),J=1,4),K,K=1,M),SL,KZU
C     FUNCTION RO$$$$(/STOFF/,/A1/,/A2/,/A3/)
 3000 FORMAT(6X,'FUNCTION ' ,A2,4A1,'(/STOFF',8(4A1,I1),
     1       4(3A1/5X,'1',A1,I2,10(4A1,I2)))
      WRITE(P,1050) DAT
C
CN    ***  M A P L I B  ***    RETRIEVAL  FUNCTION
C
CA    MAPLIB-UTILITY
CD    07.07.70
      WRITE(P,1052)(ERKEIG(J,I),J=1,17)
C$P   DICHTE                                                    KG/M**3
 1052 FORMAT('C$P',3X,17A4)
      IF (L.GT.1) GOTO 17
      WRITE(P,3100) KATEIG(I),L,STOKAT(1),AP,SL
      WRITE(P,3200) M,L,KZU
      GOTO 3201
   17 WRITE (P,3100) KATEIG(I),L,STOKAT(1),((TRA(J),J=1,3),
     1                STOKAT(K),K=2,L),AP,SL
C     COMMON /$NUMB$/ J,LJ
C     LOGICAL*1 LJ
C     INTEGER VALUE(2)/'RO    '/,STOFF,STOKAT( 11)/'4988','4981',
C    1'4961','UPUO','PUO ','UO  ','B4O ','NALS','NAV ','NAL ','NAVS'/
C    1
 3100 FORMAT(6X,'COMMON /$NUMB$/ J,LJ'/6X,'LOGICAL*1 LJ'
     1      /6X,'INTEGER VALUE(2)/''',A2,'    ''/',STOFF,STOKAT(',I3,
     2       ')/''',A4,3A1,A4,11(2A1/5X,'1',A1,A4,8(3A1,A4)))
   18 WRITE (P,3200) M,L,(TRA(2),J,J=2,L),KZU
C     IF (LJ) J=NUMBR$( 3)-1
C     LJ=.FALSE.
C     DO 9998 I=1, 11
C     IF (STOFF.EQ.STOKAT(I)) GOTO(1, 2, 3, 4, 5, 6, 7, 8, 9,10,11)
C    1
 3200 FORMAT(6X,'IF (LJ) J=NUMBR$(',I2,')-1'/6X,'LJ=.FALSE.'
     1      /6X,'DO 9998 I=1,',I3/6X,'IF (STOFF.EQ.STOKAT(I)) GOTO(1',
     2       10(A1,I2),4(A1/5X,'1',I2,20(A1,I2)),A1/5X,'1',I2,4(A1,I3))
 3201 CONTINUE
      WRITE (P,3300) KATEIG(I),$,$,$,$
C    1 ,I
```

```
C9998 CONTINUE
C     VALUE(2)=STOFF
C     RO$$$$=ERRX$$(VALUE)
C9999 LJ=.TRUE.
C     RETURN
 3300 FORMAT(    5X,'1 ,I'/' 9998 CONTINUE'/6X,'VALUE(2)=STOFF'
     1      /6X,A2,4A1,'=ERRX$$(VALUE)'/' 9999 LJ=.TRUE.'/6X,'RETURN')
      NRF11=21+(M+2)/11+(L+ 7)/ 9+(L+10)/21+2*L+NRF11
      DO 31 J=1,L
      K=NEIG(J)
      EIGKAT(J)=KATEIG(I)
      IF(.NOT.LSYN(K)) GOTO 31
      KK=NRSYN(K)
      EIGKAT(J)=SYNEIG(KK)
      STOKAT(J)=SYNSTO(KK)
   31 CONTINUE
      DO 14 J=1,L
      NN=NARG(NEIG(J))
      IF (NN .GT. 1) GOTO 16
      WRITE(P,3400) J,KATEIG(I),$,$,$,$,EIGKAT(J),STOKAT(J)
C    5 RO$$$$=RONAV (A1,A 2)
 3400 FORMAT(2X,I3,' ',A2,4A1,'=',A2,A4,'(A1)')
      GOTO 15
   16 WRITE(P,3500) J,KATEIG(I),$,$,$,$,EIGKAT(J),STOKAT(J)
     1                 ((TRK(K),K=1,2),N,N=2,NN),KZU
 3500 FORMAT(2X,I3,' ',A2,4A1,'=',A2,A4,'(A1',12(2A1,I2),
     1       3(/5X,'1',16(2A1,I2)))
   15 WRITE (P,1500)
C     GO TO 9999
      NRF11=NRF11+(NN+3)/16
   14 CONTINUE
      WRITE(P,1600)
C     END
```

```
      NREIGE(I)=NRF11
   12 CONTINUE
      CALL END11(P)
      RETURN
      END
```

```
      SUBROUTINE  FILCUT(NF,IR1,IR2,TEXT,IT)
CN    FILCUT   AUSGABE-ROUTINE
C
CB    AUSGABE DES RECORDS IR1 BIS IR2 VON DER EINHEIT NF AUF DIE EINHEIT
CB    OUT SOWIE BEI  PUNCH=.TRUE. AUF DIE EINHEIT PRT.
CB    BEI LINK=.TRUE. AUSGABE DER JCL MIT EXEC1/2.
CP    TEXT(IT)  BEI DER AUSGABE AUF OUT WIRD DER TEXT ALS UEBERSCHRIFT
CP    GEDRUCKT
C
      COMMON/SKCNT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JFIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      COMMON/SKCNT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1    ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,  FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 TEXT(IT),LINK,NOEND,LA(80),LN(8),LALIAS(76),LINKO,
     1 PUNCHO,LEQ,LBLK
     2,LNOC,L$$
      INTEGER*2 ZAHL(7)/' 1 2 3 4 5 6 0'/,A2,CE/'$$'/,BLK2,NAL(2)
     1,EQU/'  '/,KL/'( '/
     2,C/'C '/
      INTEGER A(20),         BLK/'    '/,NXX,NYYYY,IPAGE/1/,N13/13/
      INTEGER C$O/'C$O '/,C$OM/'C$OM'/,C$/'    '/
      REAL*8 AEND,ENDB/'  END   '/,SUBR/'  SUBROU'/
      DIMENSION IAEND(2)
      DATA NFO/0/,IRO/1/
      EQUIVALENCE(A(2),LALIAS(1)),(NURSOR,LINK),(BLK,BLK2,LBLK),
     1 (NXX,LN(1)),(NYYYY,LN(5)),(A(1),LA(1),A2,NAL(1))
     2,(AEND,IAEND(1)),(EQU,LEQ)
     3,(LNOC,NOTDOU)
      LINKO=LINK
      PUNCHO=PUNCH
      L$$=.FALSE.
      NREC=IR2-IR1+1
      NOEND=.FALSE.
      IF(NF.EQ.9) GOTO 20
      IF(NF.NE.NFO) GOTO 18
      IF(IR1-IRO)18,17,19
C
CB    BEI SEQUENTIELLEN DATA-SETS WIRD IR1 DURCH REWIND UND UEBERLESEN
CB    DER IR1-1 RECORDS GEFUNDEN. FUER EINEN FOLGENDEN AUFRUF FUER DIE
CB    GLEICHE EINHEIT NF=NFO WIRD DER ZULETZT GELESENE RECORD IN IRO
CB    GEMERKT, SODASS BEI IR1>IRO KEIN NEUES REWIND ERFORDERLICH IST.
C
   18 REWIND NF
      IRO=1
      IF(IR1.EQ.1) GOTO 17
   19 J=IR1-IRO
      DO 16 I=1,J
   16 READ(NF,100,END=2,ERR=4) NXX
      GOTO 17
   20 IAA=IR1
   17 J=1
      DO 10 I= 1,NREC
      IF(NF-NDA) 40,41,40
```

```
CB      BEI NF .NE. NDA WIRD SEQUENTIELL GELESEN
CB      BEI NF .EQ. NDA WIRD MIT GETREC VON NDA DIRECT GELESEN
   40 READ(NF,100,END=2) A
      IF(.NOT.LNCC) GOTO 42
      LEQ=LA(1)
      IF(EQU.EQ.C.AND.A(1).NE.C$O.AND.A(1).NE.C$OM) GOTO  10
      GOTO 42
   41 CALL GETREC(A,IAA)
      IAA=IAA+1
      IF(.NOT.LNCC) GOTO 42
      LEQ=LA(1)
      IF(EQU.EQ.C.AND.A(1).NE.C$O.AND.A(1).NE.C$OM) GOTO  10
   42 IAEND(1)=A(2)
      IAEND(2)=A(3)
      IF(MOD((J-1),60).NE.C  .AND. NOEND) GOTO 321
      IF(L$$) GOTO 321
      IF(NOEND)GOTO43
      J=1
      IF(LINK)CALL EXEC1(PRT)
      GOTO 44
   43 IF(J.GT.10) WRITE(OUT,105)
   44 WRITE(OUT,103) TEXT
  103 FORMAT('1MAPLIB-UTILITY',7X,8CA1)
      WRITE(OUT,104) IPAGE
  104 FORMAT('+',T120,'PAGE',I4//)
      IPAGE=IPAGE+1
      NOEND=.TRUE.
      WRITE(OUT,105)
  105 FORMAT(15X,82('$'))
      IF(.NOT.NOTSTO) GOTO 320
      IF(.NOT.LINK) GOTO 12
  320 IF(J.GT.10) GOTO 321
      N13=12
      IF(AEND.EQ.SUBR) N13=14
  303 LEQ=LA(N13+3)
      IF(EQU.NE.BLK2) GOTO 302
      N13=N13+1
      GOTO 303
  302 DO 304 K=2,8
  304 LN(K)=LBLK
      LN(1)=LEQ
      LEQ=LA(N13+4)
      IF(EQU.EQ.KL) GOTO 305
      LN(2)=LEQ
      DO 23 K=5,8
      LEQ=LA(K+N13)
      IF(EQU.EQ.KL) GOTO 305
   23 LN(K)=LEQ
  305 LN(3)=LN(5)
      LN(4)=LN(6)
  321 L$$=A2.EQ.CE
      IF(L$$)GOTO 1
      IF(NOTSTO) GOTO 12
      A(19)=NXX
      A(20)=NULLZZ(K)
   12 WRITE(OUT,101) J,A
      J=J+1
```

```
  101 FORMAT(' CARD',I5,5X,'$',20A4,'$')
  100 FORMAT(20A4)
      IF(PUNCH) WRITE(PRT,100)A
      NCEND=AEND.NE.ENDB
      IF(.NOT.  NCEND) NCEND=(C$.EQ.C$O) .OR. (C$.EQ.C$CM)
      C$=A(1)
    1 IF(.NOT.LINK) GOTO 5
      IF(.NOT.L$$ .AND. NCEND) GOTO 10
      LINK=.FALSE.
      K=0
      IF(.NOT.NCEND) GOTO 11
      WRITE(PRT,106) A(19),A(20)
  106 FORMAT(6X,'END',T73,2A4)
      PUNCH=.FALSE.
      IF(NAL(2).EQ.ZAHL(7).OR.NAL(2).EQ.BLK2) GOTO 11
      DO 32 K=1,6
      IF(NAL(2).EQ.ZAHL(K)) GOTO 11
   32 CONTINUE
   11 CALL EXEC2(PRT,NXX,NYYYY,K,LALIAS)
    5 IF(NCEND) GOTO 10
      J=0
      WRITE(OUT,105)
      LINK=LINKO
      PUNCH=PUNCHO
   10 CONTINUE
      IF(NCEND) WRITE(OUT,105)
      NFO=NF
      IRO=IR2+1
    3 RETURN
C
      ENTRY PAGE
CN    PAGE
CB    INITIALISIERUNG DES PAGE-ZAEHLERS IPAGE
      IPAGE=1
      RETURN
C
      ENTRY INIT
CN    INIT
CB    INITIALISIERUNG DER MERK-VARIABLEN NFO UND IRO ZU BEGINN ODER
CB    BEI I/O  DIESER DATA-SETS AN ANDERER STELLE.
      NFO=0
      IRO=0
      RETURN
    2 CALL FEHLER('NREC WRONG * FILOUT',19,83)
    4 WRITE(6,102) NF,NREC,I
  102 FORMAT('0***ERROR IN FILOUT : NF=',I3,5X,'NREC=',I6,5X,'I=',I6)
      GO TO 3
      END
```

```
      FUNCTION NULLZZ(/DUMMY/)
      LOGICAL*1 LZ(4)/'0000'/,LZO(11)/'0123456789 '/
      INTEGER*2 KZ(4)/1,1,1,1/
      EQUIVALENCE (NULL,LZ(1))
CN    NULLZZ- ZIFFERN-KETTE MIT VORHERGEHENDEN NULLEN
CA    SCHUMANN
CC    15.06.70
CB    DIE FUNKTION LIEFERT BEIM ERSTEN AUFRUF DIE ZEICHENKETTE '0001',
CB    BEIM  2.: '0002', ... ,'9999','0000','0001'     USW.
CP    DAS ARGUMENT IST EIN DUMMY-PARAMETER, DER NICHT VERAENDERT WIRD
      DO 300 M=1,4
      K=5-M
      KZ(K)=KZ(K)+1
      I=KZ(K)
      LZ(K)=LZO(I)
      IF(I.LT.11) GOTO 301
      KZ(K)=1
  300 LZ(K)=LZO(1)
  301 NULLZZ=NULL
      RETURN
      END
```

```
      SUBROUTINE EXEC1(N)
CN    EXEC1
CB    AUSGABE DER EXEC- UND DER DD-KARTEN SOWIE DER LINKAGE-EDITOR-STEU
CB    ERKARTEN FUER DIE ERZEUGUNG VON LOAD-MODULES
CB    DIE JOB-KARTE WIRD MIT HILFE VON JOBCD1 GESCHRIEBEN
CP    N    AUSGABE-EINHEIT
C
      LOGICAL*1 L(7)/'       '/,LE(2,6),LB/' '/,LR(3)/'(R)'/,KOM/','/,
     1LA(44),ALIAS(6,1)
      INTEGER*2 AA(6)/'            '/,BLK2/'  '/
      INTEGER A,Y,STEP/0/,DSN/1/
     1 , X
      INTEGER COMP, COMPL(4)/'FH2 FH1 FH0 FG  '/,C/1/,CM/4/,C2/2/
     1 ,DSNAME,DSNAM(2)/'LOAD','MAPL'/,EXEC(2)/'EXEC'/
      INTEGER ITIME(2)/1,5/
      EQUIVALENCE (L(1),A),(LE(1,1),AA(1))
      IF (STEP.NE.0) GOTO 11
      CALL JOBCD1(N)
   11 EXEC(2)=NULLZZ(CM)
      GOTO(71,72),DSN
   71 GOTO(711,711,711,714),C
  711 WRITE(N,100) ITIME,C2,EXEC
  100 FORMAT('//MAPLIB EXEC LFHCL,LIB=IRE,TIME=(,',2I1,'),PARM.C=''OPT=
     1,I1,''',PARM.L=''NCAL''',T73,2A4)
      GOTO 800
  714 WRITE(N,200) ITIME,EXEC
  200 FORMAT('//MAPLIB EXEC LFGCL,LIB=IRE,TIME=(,',2I1,'),PARM.L=''NCAL
     1'',T73,2A4)
  800 EXEC(2)=NULLZZ(C2)
      WRITE(N,500) EXEC
  500 FORMAT('//C.SYSIN DD *',T73,2A4)
      GOTO 20
   72 GOTO(721,721,721,724),C
  721 WRITE(N,110) ITIME,C2,EXEC
  110 FORMAT('//MAPLIB EXEC FHCL,TIME=(,',2I1,'),PARM.C=''OPT=',I1,''',P
     1ARM.L=''NCAL''',T73,2A4)
      GOTO 800
  724 WRITE(N,210) ITIME,EXEC
  210 FORMAT('//MAPLIB EXEC FGCL,TIME=(,',2I1,'),PARM.L=''NCAL''',T73,
     1 2A4)
      GOTO 800
C
      ENTRY EXEC2(N,X,Y,NAL,ALIAS)
CP    X    EIGENSCHAFTSSYMBOL-TEIL DES ROUTINEN-NAMENS
CP    Y    STOFFSYMBOL        -TEIL DES ROUTINEN-NAMENS
CP    NAL  ZAHL DER ENTRIES FUER DIE ALIAS-ANWEISUNG
CP    ALIAS NAMEN DER ENTRIES
      EXEC(2)=NULLZZ(N)
      WRITE(N,101) EXEC
  101 FORMAT('/*',T73,2A4)
      EXEC(2)=NULLZZ(C2)
      IF(DSN.EQ.1) GOTO 900
      WRITE(N,211) EXEC
  211 FORMAT('//L.SYSLMOD DD DSN=MAPLIB.IRE.LOAD,SPACE=,UNIT=2314,DISP=(
     1LD,',T73,2A4)
      EXEC(2)=NULLZZ(N)
      WRITE(N,212) EXEC
```

```
  212 FORMAT('// VOL=SER=GFK006',T73,2A4)
      EXEC(2)=NULLZZ(I)
  900 WRITE(N,105) EXEC
  105 FORMAT('//L.SYSIN DD *',T73,2A4)
      IF(NAL.EQ.   0) GOTO 5
      K=1
      DO6I=1,44
    6 LA(I)=LB
      DO 7 J=1,NAL
      DO 8 I=1,6
      LE(1,I)=ALIAS(I,J)
      IF(AA(I).EQ.BLK2) GOTO 9
    8 CONTINUE
      I=7
    9 I=I-1
      K2=K+I-1
      I=1
      DO10M=K,K2
      LA(M)=ALIAS(I,J)
   10 I=I+1
      IF(J.LT.NAL)LA(K2+1)=KOM
      K=K2+2
    7 CONTINUE
      EXEC(2)=NULLZZ(K)
      WRITE(N,103) LA,EXEC
  103 FORMAT(' ALIAS ',44A1,T73,2A4)
    5 A=Y
      DO1I=1,5
      LE(1,I)=L(I)
      IF(AA(I).EQ.BLK2) GOTO 2
    1 CONTINUE
    2 I=I-1
      DO 3 J=1,3
    3 L(J+I)=LR(J)
      EXEC(2)=NULLZZ(STEP)
      WRITE(N,102) X,L,EXEC
  102 FORMAT(' NAME ',A2,7A1,T73,2A4)
      EXEC(2)=NULLZZ(I)
      WRITE(N,101) EXEC
      DO 4 I=5,7
    4 L(I)=LB
      STEP=STEP+1
      IF(STEP.LT.7) RETURN
      STEP=0
      EXEC(2)=NULLZZ(STEP)
      WRITE(N,104)
  104 FORMAT('//',76X,'//')
      RETURN
C
      ENTRY EXEC3(COMP)
      DO 40 I=1,CM
      IF(COMP.EQ.COMPL(I)) GOTO 41
   40 CONTINUE
      C=1
      C2=2
      CALL FEHLER('MAPLIB DOES NOT KNOW THIS COMPILER; FH2 IS ASSUMED',
     150,&20)
```

```
   41 C=I
      C2=3-I
      GOTO 20
C
      ENTRY EXEC5(IN)
      READ(IN,1200,ERR=1201) ITIME
 1200 FORMAT(2I1)
      GOTO 20
 1201 CALL FEHLER('ERROR WHILE READING NEW TIME'        ,28,&1202)
 1202 ITIME(1)=1
      ITIME(2)=5
      GOTO 20
C
      ENTRY EXEC4(DSNAME)
      DO  50  I=1,2
      IF(DSNAME.EQ.DSNAM(I)) GOTO 51
   50 CONTINUE
      DSN=1
      CALL FEHLER('MAPLIB DOES NOT KNOW THIS DSNAME; LOAD.IRE IS ASSUMED
     1',53,&20)
   51 DSN=I
   20 RETURN
      END
```

```
      SUBROUTINE JCECDO(IN)
CN    JOBCDO   VERARBEITEN DER JCB-CARD
CB    JCBCDO LIEST VON DER EINHEIT  IN DIE AUSZUGEBEND JOB-CARD EIN.
CB           SIE ENTHAELT JEDOCH AUCH EINE STANDARD-KARTE.
C     WITH THIS ROUTINE IT IS POSSIBLE TO PUNCH 1296 DIFFERENT
C     JOBCARDS
CP    IN    EINGABE-EINHEIT
      LOGICAL*1 ABC(36)/'0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ'/,Q(80)
     1/'//IRE67300 JOB (0673,330,POOOO),SCHUMANN,MSGLEVEL=(1,1),CLASS=A
     2                 '/
      LOGICAL*1 CARD2/.FALSE./,LQ(2,80)  ,Q2(80)
      INTEGER*2 QE(80)/80*'  '/,JOB(5)/'  J O B   '/,KOM/', '/,BLK2
      INTEGER*2 MZ(2),X/'  '/
      LOGICAL*1 LX
      EQUIVALENCE (LQ(1,1),QE(1)),(JOB(1),BLK2)
      EQUIVALENCE(LX,X)
      READ(IN,100,END=1001,ERR=1002) QE
  100 FORMAT( 80A1)
      DO 1 I=10,67
      DO 2 K=1,5
      IF(QE(I+K).NE.JOB(K)) GO TO 1
    2 CONTINUE
      GOTO 3
    1 CONTINUE
      CALL FEHLER('THE KEY ''JOB'' COULD NOT BE FOUND',32,&10)
    3 K=I+5
      DO 11 I=K,72
      IF(QE(I).NE.BLK2) GOTO 12
   11 CONTINUE
      CALL FEHLER('THE JOB-CARD IS EMPTY',21,&10)
   10 CALL FEHLER('STANDARD JOB-STATEMENT WILL BE USED',35,&20)
   12 K=I+1
      DO 5 I=K,73
      IF(QE(I).EQ.BLK2) GOTO 4
    5 CONTINUE
      CALL FEHLER('NO BLANK ON JOB-CARD BEFORE COLUMN 73',37,&10)
    4 CARD2= QE(I-1).EQ.KOM
      IF(CARD2) READ(IN,100,END=1001,ERR=1002) Q2
      DO 6 I=3,80
    6 Q(I)= LQ(1,I)
   20 RETURN
 1002 CALL FEHLER('ERROR WHILE READING JCBCARD',27,&10)
 1001 CALL FEHLER('THERE IS NO JOB-CARD',20,&20)
      ENTRY JCBCD1(OUT)
CP    OUT   AUSGABE-EINHEIT
CN    JCBCD1
CB    AUSBABE DER JCB-KARTE MIT FORTLAUFENDER NUMMERIERUNG
      INTEGER OUT,NR/0/
      IF(NR.GT.1295) NR=0
      K1=NR/36
      K2=NR-K1*36+1
      K1=K1+1
      I=NULLZZ(I)
      IF(CARD2)I=NULLZZ(I)
      WRITE(OUT,101)(Q(I),I=3,8),ABC(K1),ABC(K2),(Q(I),I=11,80)
  101 FORMAT('//',78A1)
      IF(CARD2)WRITE(OUT,101)(Q2(I),I=3,80)
```

```
      NR=NR+1
      RETURN
      ENTRY JOBCD2(IN)
CN    JOBCD2
CB    EINLESEN DER ANFANGSNUMMER; STANDARD: 00
      READ(IN,100) MZ
      NR=0
      DO 200 I=1,36
      LX=ABC(I)
      IF(MZ(1).EQ.X)NR=NR+(I-1)*36
      IF(MZ(2).EQ.X)NR=NR+I-1
  200 CONTINUE
      GOTO 20
C     RETURN
      END
```

```
      SUBROUTINE TABWRT
CN    TABWRT - SCHREIBEN DER TABELLE
CB    AUSGABE GEMAESS ANWEISUNG  INFO/.TAB
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTOA(100),NRSTOE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON/SDIM2/ EIG(500),NARG(500),KATEIG(100),SYNEIG(20),KATARG
     1(3,50),SYSTEM(3,50)
      INTEGER*2 EIG,NARG,KATEIG,SYNEIG,KATARG,SYSTEM
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      DIMENSION KST(1)
      INTEGER*2 AE,KEIG(1)
      INTEGER A(30),BLK/'    '/,AS,STERN/' *  '/
      EQUIVALENCE(KS,NST),(KE,NEIG),(KATSTO(1),KST(1)),
     1(KATEIG(1),KEIG(1))
C
CB    AUSGABE DER FUNCTION-TABELLE
C
  998 WRITE(OUT,100)
  100 FORMAT('1***  TABLE OF THE CATALOGUED FUNCTIONS'/ 9X,
     1'MARKED BY THEIR PARAMETER-LIST'//)
      IF(KS.LE.0.OR.KE.LE.0.OR.JEIN.LE.0)CALL FEHLER(
     1   'THE CATALOGUE IS EMPTY',22,8999)
      NSTO=1
      NST1=5
    7 IF(NST.LT.NST1) NST1=NST
      NEIGO=1
      NEIG1=25
    5 IF(NEIG.LT.NEIG1)NEIG1=NEIG
      WRITE(OUT,101)(KST(I),I=NSTO,NST1)
  101 FORMAT(' PROPERTY     /  MATERIAL'/12X,A4,4(20X,A4)/)
      DO 4 K=NEIGO,NEIG1
      DO 1 I=1,30
    1 A(I)=BLK
      L=6*(MOD((NST1-1),5)+1)
      DO 8 I=1,L,6
    8 A(I)=STERN
      AE=KEIG(K)
      M1=1
      M2=6
      DO 2 I=NSTO,NST1
      AS=KST(I)
      DO 35 L=1,JEIN
      IF(EIG(L).NE.AE)GOTO 35
      IF(STO(L).NE.AS)GOTO 35
      GOTO 31
   35 CONTINUE
      GOTO 34
   31 DO 33 LL = M1,M2
   33 A(LL)=ARG(LL-M1+1,L)
   34 M1=M1+6
      M2=M2+6
```

```
    2 CONTINUE
      WRITE(OUT,102) KEIG(K),(A(I),I=1,30)
  102 FORMAT('0',7X,A2,2X,30A4)
    4 CONTINUE
      IF(NEIG1.EQ.NEIG) GOTO 6
      NEIG0=NEIG1+1
      NEIG1=NEIG1+25
      WRITE(OUT,103)
  103 FORMAT('1PROPERTIES CONTINUED'/)
      GO TO 5
    6 IF(NST1.EQ.NST) GO TO 2999
      NST0=NST1+1
      NST1=NST1+5
      WRITE(OUT,104)
  104 FORMAT('1MATERIALS CONTINUED'/)
      GO TO 7
 2999 CONTINUE
      WRITE(OUT,200)
  200 FORMAT(
     1'0* EXPLANATIONS'/
     2'0IF THERE EXISTS A PARAMETER-LIST, THE FUNCTION DEFINED BY THE CO
     3MBINATION OF THE PROPERTY- AND MATERIAL-SYMBOL IS INTEGRATED IN'/
     4' MAPLIB AND CAN BE CALLED WITH THE PARAMETERS SHOWN.'/
     5'0IF THERE IS ONLY A ''*'', THE FUNCTION IS NOT INTEGRATED IN MAPL
     6IB; PLEASE ADD. '/
     7'0IF THE PARAMETER-LIST CONSISTS OUT OF A LOT OF PARAMETER-SYMBOLS
     8 WITH ONLY ONE CHARACTER OR DOES NOT END WITH A PARENTHESIS,'/
     9' THE PARAMETER-SYMBOLS ARE ABBREVIATED; REFER TO THE FUNCTION-LIS
     1T FOR THE COMPLETE PARAMETER-LIST.')
      WRITE(OUT,203)
  203 FORMAT(
     1'0IF IN THE PARAMETER-LIST A '','' IS SUBSTITUTED BY A '';'',
     2'', THE PARAMETERS AFTER THE '';'' ARE OPTIONAL.')
      IF(KY.GT.0) WRITE(OUT,201)
     1 (I,EIG(JSYN(I)),STO(JSYN(I)),SYNEIG(I),SYNSTO(I),I=1,KY)
  201 FORMAT(
     1'0IF THE PARAMETER-LIST STARTS WITH A ''$'' INSTEAD OF THE LEFT PA
     2RENTHESIS, THE FUNCTION MUST BE CALLED BY A SYNONYM-NAME;'/
     3' REFER TO THE FOLLOWING LIST FOR SYNONYM-NAMES.'/
     4'1** LIST OF SYNONYM-NAMES'//
     5'0',T9,'FUNCTION',T25,'SYNONYM'// (' ',I3,T9,A2,A4,T25,A2,A4))
      IF(KY.EQ.0) WRITE(OUT,202)
  202 FORMAT('0** NO SYNONYM-NAMES ARE DEFINED')
 3000 FORMAT('1**  LIST OF INTEGRATED SYSTEM ROUTINES'//)
      IF(K$.LE.0) GOTO 4000
      DO 3001 I=1,K$
      IF(MOD((I-1),50).EQ.0)WRITE(OUT,3000)
 3001 WRITE(OUT,3002) I,(SYSTEM(J,I),J=1,3)
 3002 FORMAT(I6,3X,3A2)
 4000 WRITE(OUT,4001) NRF11
 4001 FORMAT('0** NUMBER OF RECORDS ON THE SOURCE FILE:',I9)
  999 RETURN
      END
```

```
      SUBROUTINE SCRA
CN    SCRA
CB    COMPRIMIERUNG DES DATA-SETS AUF DER SOURCE-EINHEIT
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTCA(100),NRSTCE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON/SKONT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1   ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,  FUNCDA,NEWMAS,PUNCH
      COMMON/SKONT4/ NDIMAX,NBRREC,NCA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      NRF11=0
      CALL F11TOS
    4 REWIND NF1
      FUNCDA=.FALSE.
C     RUL
C     MES
      CALL SCCOMP(JEIN,NRECA,NRECE)
      CALL SCCOMP(K$,NRECSA,NRECSE)
      CALL SCCOMP(1,N$$A,N$$E)
      CALL SCCOMP(KS,NRSTCA,NRSTCE)
      CALL SCCOMP(KE,NREIGA,NREIGE)
      CALL END11(NF1)
      WRITE(OUT,1010) NF1
 1010 FORMAT('0*** FILE',I3,' COMPRESSED')
      RETURN
      END
```

```
      SUBROUTINE  SCCOMP(JEIN,NRECA,NRECE)
CN    SCCOMP
CB    HILFSROUTINE ZU SCRA
      COMMON /SKONT4/ ICOMMO(2),NDA,IDUMM1(18),NF1,IDUMM2(3),NREF11,IAV,
     1IDUMM3(5)
      DIMENSION NRECA(JEIN),NRECE(JEIN),A(20)
      IF(JEIN.EQ.0) RETURN
      DO 1  J=1,JEIN
      IAA=NRECA(J)
      N= NRECE(J)-NRECA(J)+1
      NRECA(J)=NREF11+1
      NREF11=NREF11+N
      NRECE(J)=NREF11
      DO 2 I=1,N
      CALL GETREC(A,IAA)
      IAA=IAA+1
    2 WRITE(NF1,1000) A
    1 CONTINUE
      RETURN
 1000 FORMAT(20A4)
      END
```

```
      FUNCTION PASSW(I)
CN    PASSW
CC    1.4.7C
CB    ZUR SICHERUNG DER DATENBESTAENDE WIRD BEI MANCHEN OPERATIONEN
CB    EIN PASSWORT ERWARTET
CB    DIESE FUNKTION LIEFERT DAS RICHTIGE PASSWORT,
CB    ES IST IN  WORT  GESPEICHERT
      REAL*8 PASSW,WORT/'$MAPLIB$'/
      PASSW=WORT
      RETURN
      END
```

```
      SUBROUTINE DELARG(*)
CN    DELARG  -DELETE/.PARAMETER
CB    LOESCHEN VON PARAMETERN AUS DEM KATALOG
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTOA(100),NRSTOE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON/SDIM2/ EIG(500),NARG(500),KATEIG(100),SYNEIG(20),KATARG
     1(3,50),SYSTEM(3,50)
      INTEGER*2 EIG,NARG,KATEIG,SYNEIG,KATARG,SYSTEM
      COMMON/SDIM1/ LSYN(500),LERL(4000),FIL(80,100)
      LOGICAL*1LSYN,LERL,FIL
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      COMMON/SKONT2/ NAMAX,NAMAXO
      INTEGER*2 NAMAX,NAMAXO
      COMMON/SKONT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1    ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,   FUNCDA,NEWMAS,PUNCH
C
C
      REAL*8 TEXT(3)/'ARGUMENT','SYSTEM  ','FUNCTION'/
      INTEGER*2 PARM(4),EIGEN
      INTEGER END/'END '/,STOFF
      EQUIVALENCE(NEND,PARM(1),EIGEN),(STOFF,PARM(3))
 1000 FORMAT(4A2)
    1 READ(INP,1000,END=997) PARM
      IF(NEND.EQ.END) GOTO 999
      CALL LSTCON(PARM,KATARG,3,1,I,KA,&998)
      WRITE(OUT,2000) TEXT(1), (PARM(J),J=1,3)
      KA=KA-1
      IF(I.GT.KA) GOTO 1
      DO 100 J= I,KA
      DO 101 K=1,3
  101 KATARG(K,J)=KATARG(K,J+1)
      DO 100 K=1,17
  100 ERKARG(K,J)=ERKARG(K,J+1)
      GOTO 1
  999 RETURN
  997 RETURN 1
  998 WRITE(6,200) (PARM(I),I=1,3)
  200 FORMAT('0*** ERROR ***   ',3A2)
      CALL FEHLER('THIS PARAMETER-NAME DOES NOT EXIST',34,&1)
C
      ENTRY DELSYS(*)
CN    DELSYS  -DELETE/.SYSTEM
CB    LOESCHEN VON SYSTEM-ROUTINEN IM KATALOG
 1001 READ(INP,1000 ,END=997) PARM
      IF(NEND.EQ.END) GOTO 999
      CALL LSTCON(PARM,SYSTEM,3,1,I,K$,&1998)
      NOTARG=.TRUE.
      WRITE(OUT,2000) TEXT(2), (PARM(J),J=1,3)
      K$=K$-1
```

```
      IF(I.GT.K$) GOTO 1001
      DO 1100 J=I,K$
      DO 1101 K=1,3
 1101 SYSTEM(K,J)=SYSTEM(K,J+1)
      NRECSA(J)=NRECSA(J+1)
 1100 NRECSE(J)=NRECSE(J+1)
      GOTO 1001
 1998 WRITE(6,200) (PARM(I),I=1,3)
      CALL FEHLER('THIS SYSTEM-NAME DOES NOT EXIST',31,&1001)
C
      ENTRY DELFUN(*)
CN    DELFUN
CB    LOESCHEN VON FUNCTIONS IM KATALOG
CB    WENN MIT DEN VERBLEIBENDEN FUNCTIONS EIN STOFF ODER EINE EIGEN-
CB    SCHAFT NICHT MEHR BESCHRIEBEN WIRD, WERDEN AUCH DIESE GELOESCHT.
 2001 READ(INP,1000,END=997) PARM
      IF(NEND.EQ.END) GOTO 2999
      PARM(4)=PARM(3)
      PARM(3)=PARM(2)
      CALL KATCON(EIG,STO,EIGEN,STOFF,1,I,JEIN,&2998)
      NOTARG=.TRUE.
      NEWMAS=.TRUE.
      JEIN=JEIN-1
      WRITE(OUT,2000) TEXT(3), EIGEN,(PARM(J),J=3,4)
 2000 FORMAT('0* ',A8,'-MEMBER *',3A2,'* HAS BEEN DELETED')
      IF(LSYN(I)) CALL DELSYN(I)
      IF(I.GT.JEIN) GOTO 2001
      DO 2100 J=I,JEIN
      EIG(J)=EIG(J+1)
      STO(J)= STO(J+1)
      DO 2101 K=1,6
 2101 ARG(K,J)=ARG(K,J+1)
      NRECA(J)=NRECA(J+1)
      NRECE(J)=NRECE(J+1)
      NARG(J)=NARG(J+1)
 2100 LSYN(J)=LSYN(J+1)
      IF(KY.EQ.0) GOTO 2001
      DO 2103 K=1,KY
      IF(I.LT.JSYN(K)) JSYN(K)=JSYN(K)-1
 2103 CONTINUE
      GOTO 2001
 2998 WRITE(6,200) PARM(1),PARM(3),PARM(4)
      CALL FEHLER('THIS FUNCTION-NAME DOES NOT EXIST',33,&2001)
      ENTRY DELCOM
CN    DELCOM - DELETE COMMENT
CB    WIRD VON PSDAT AUFGERUFEN, UM STOFF- ODER EIGENSCHAFTS-SYMBOLE,
CB    DIE NUR MIT COMMENT, OHNE FUNKTIONEN INTEGRIERT WURDEN ZU LOESCHEN
 2999 I=1
      IF(JEIN.EQ.0) GOTO 3300
 3012 IF(I.GT.KS) GOTO 3301
 3010 DO 3001 J=1,JEIN
      IF(KATSTO(I).EQ.STO(J)) GOTO 3000
 3001 CONTINUE
      WRITE(OUT,300) KATSTO(I)
  300 FORMAT('0* MATERIAL *'A4,'* IS NO LONGER DESCRIBED AND HAS BEEN DE
     1LETED THEREFORE')
      KS=KS-1
```

```
      IF(I.GT.KS) GOTO 3301
      DO 3003 J=I,KS
      KATSTC(J)=KATSTC(J+1)
      NRSTCA(J)=NRSTCA(J+1)
      NRSTCE(J)=NRSTCE(J+1)
      DO 3003 K=1,17
      ERKSTC(K,J)=ERKSTC(K,J+1)
3003  CONTINUE
      GOTO 3012
3000  I=I+1
      GOTO 3012
3300  KS=0
      KE=0
      KA=0
      NAMAX=0
      N$$A=0
      N$$E=0
      WRITE(OUT,301)
 301  FORMAT('0*** ALL FUNCTIONS, MATERIALS, PROPERTIES AND ARGUMENTS HA
     1VE BEEN DELETED')
      GOTO 999
3301  I=1
3112  IF(I.GT.KE) GOTO 3201
3110  DO 3101 J=1,JEIN
      IF(KATEIG(I).EQ.EIG(J)) GOTO 3100
3101  CONTINUE
      KE=KE-1
      WRITE(OUT,302) KATEIG(I)
 302  FORMAT('0* PROPERTY *',A2,'* IS NO LONGER DESCRIBED AND HAS BEEN D
     1ELETED THEREFORE')
      IF(I.GT.KE) GOTO 3201
      DO 3103 J=I,KE
      KATEIG(J)=KATEIG(J+1)
      NREIGA(J)=NREIGA(J+1)
      NREIGE(J)=NREIGE(J+1)
      DO 3103 K=1,17
      ERKEIG(K,J)= ERKEIG(K,J+1)
3103  CONTINUE
      GOTO 3112
3100  I=I+1
      GOTO 3112
3201  NAMAX=0
      DO 3212 J=1,JEIN
      IF(NARG(J).GT.NAMAX) NAMAX=NARG(J)
3212  CONTINUE
      GOTO 999
      END
```

```
      SUBROUTINE DELSYN(J)
CN    DELSYN  -  DELETE SYNONYM
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTOA(100),NRSTOE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON/SDIM2/ EIG(500),NARG(500),KATEIG(100),SYNEIG(20),KATARG
     1(3,50),SYSTEM(3,50)
      INTEGER*2 EIG,NARG,KATEIG,SYNEIG,KATARG,SYSTEM
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
CB    DELSYN LOESCHT DAS ZUR FUNCTION J GEHOERIGE SYNONYM (LSYN(J)=T)
      JY=NRSYN(J)
      KY=KY-1
      WRITE(OUT,100) SYNEIG(JY),SYNSTO(JY),EIG(J),STO(J)
  100 FORMAT('0* THE OLD SYNONYM *',A2,A4,'* TO *',A2,A4,'* HAS BEEN DEL
     1ETED')
      IF(JY.GT.KY) GOTO 1
      DO 2 K=JY,KY
      SYNSTO(K)=SYNSTO(K+1)
      SYNEIG(K)=SYNEIG(K+1)
    2 JSYN(K)=JSYN(K+1)
    1 RETURN
      END
```

```
      SUBROUTINE $NAME(INP,OUT,NEWMAS,*)
CN    $NAME - SPEZIFIKATION DES MASTER-ZEICHENS
C     WIRD AUFGERUFEN NACH DEM HAUPTSCHLUESSELWORT NAME
CD    1.5.7C
CE    ALTES ZEICHEN IST IN  $ALT GESPEICHERT; STANDARD: $
CB    NEUES ZEICHEN WIRD IN $NEU  EQ $$ EINGELESEN
CB    TEST CB $NEU EQ $$ EIN ZULAESSIGES ZEICHEN IST
CB    WENN JA, DANN WIRD  $ALT=$NEU
CB    WENN NEIN, DANN WIRD $ALT=$NEU=DOLLAR
CB    ALTES UND NEUES ZEICHEN WERDEN AUSGEDRUCKT
      INTEGER OUT
      INTEGER*2 $$/'$ '/,CHAR(21)/'$ A B C D E F G H O P Q R S T U V W X
     1 Y Z '/
      LOGICAL*1 DOLLAR/'$'/,$NEU,$ALT/'$'/,NEWMAS,$
      EQUIVALENCE($$,$NEU)
      CALL PUBLIC('NAME    ')
      READ(INP,1000,END=999)$NEU
 1000 FORMAT(A1)
      DO 10 I=1,21
      IF($$.EQ.CHAR(I)) GOTO 102
   10 CONTINUE
                        CALL FEHLER('THE NEW CHARACTER IS NOT ALLOWED',32,
     2&103)
  102 WRITE(OUT,101) $ALT,$NEU
  101 FORMAT('0*** THE OLD CHARACTER *',A1,'* FOR THE NAMES OF THE MASTE
     1R FUNCTIONS HAS BEEN CHANGED IN *'A1,'*')
      $ALT=$NEU
      NEWMAS=.TRUE.
      RETURN
  103 $NEU=DOLLAR
      GOTO 102
      ENTRY ZEICH($)
CN    ZEICH
CB    BEI AUFRUF VON ZEICH WIRD IN $ DAS NEUESTE ZEICHEN $NEU EINGESPEI-
CB    CHERT
CB    ZEICH  WIRD AUFGERUFEN  IN  INFO  UND VOR ERSTELLUNG DER MASTER-
CB    FUNKTIONEN IN   MAIN
      $=$NEU
      RETURN
  999 CALL FEHLER('END OF INPUT * NAME',19,&998)
  998 RETURN 1
      END
```

```
      SUBROUTINE INFO(WORT,I,*)
CN    INFO
CB    INFO ENTSCHEIDET, OB IN WORT EIN HAUPTSCHLUESSELWORT (Z.B. LINK)
CB    ODER EIN FOLGESCHLUESSELWORT MIT 1 PUNKT (Z.B..FUN)
CB    ODER EIN FOLGESCHLUESSELWORT MIT 2 PUNKTEN (Z.B. ..ES$$$$)
CB    ENTHALTEN IST.
CB    REAKTION  AN HAND FOLGENDER BEISPIELE ERLAEUTERT:
CB    LINK      ->  RETURN 1  , I UNVERAENDERT
CB    .FUN      ->  RETURN    , I UNVERAENDERT
CB    ..ES$$$$  ->  I>=1 , RETURN
CB    I=1       ->  OBERSTE MASTERFUNKTION  $$$$$$
CB    I=2       ->  STOFFFFUNCTION          $$****
CB    I=3       ->  EIGENSCHAFTS-FUNCTION   **$$$$
CB    I=4       ->  DATEN-FUNCTION          ******
      INTEGER*2 W(8)/8*'  '/,P/'. '/,$$/'$ '/
      LOGICAL*1 WORT(8),WO(2,8),EIG,STO,$
      EQUIVALENCE(W(1),WO(1,1)),($,$$)
      DO 1 J=1,8
    1 WO(1,J)= WORT(J)
      IF(W(1).NE.P) RETURN 1
      IF(W(2).NE.P) RETURN
      CALL ZEICH($)
      EIG=.FALSE.
      STO=.FALSE.
      IF(W(3).EQ.$$.AND.W(4).EQ.$$) EIG=.TRUE.
C     $$ABCD
      DO 2 J=5,8
      IF(W(J).NE.$$) GOTO 3
    2 CONTINUE
      STO=.TRUE.
    3 IF(EIG.AND.STO) I=1
      IF(EIG.AND..NOT.STO) I=2
      IF(STO.AND..NOT.EIG) I=3
      IF(.NOT.EIG.AND..NOT.STO) I=4
      RETURN
      END
```

```
      SUBROUTINE  $UNIT(*)
CN    $UNIT
CB    AENDERUNG VON I/O-EINHEITEN NACH KEY UNIT
CV    IPASSO = PASSWORT FUER DIE EINHEITEN NDA,NF1,NF3
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
C     SKONT4
      INTEGER WORT,SW(5)/'.PRI','.PUN','.INP','.SOU','.REG'/,END
     1 /'END '/,ISW/5/,IPASSO/'$MAP'/
      LOGICAL *1 STERN /'*'/
      CALL PUBLIC('UNIT    ')
      INPO=INP
    1 READ(INPO,1000,END=999) WORT
      IF(WORT.EQ.END) GOTO 50
 1000 FORMAT(A4)
      DO 3 K=1,ISW
      IF(WORT.EQ.SW(K)) GOTO 4
    3 CONTINUE
      WRITE(OUT,5) WORT,(STERN,SW(I),STERN,I=1,ISW)
    5 FORMAT('0KEW WORD  *',A4 ,'* UNKNOWN'/' PERMISSIBLE KEY WORDS ARE'
     1 ,10(2X,A1,A4,A1,2X))
   52 BACKSPACE INPO
      INP=INPO
   50 WRITE(6,6) INP,OUT,PRT,NF1,NF3
    6 FORMAT('0NEW UNITS'/'0INPUT:',I4,', PRINT:',I4,', PUNCH:',I4,
     1', SOURCE:',I4,', REGISTER:',I4)
      RETURN
    4 READ(INPO,1001,END=999,ERR=998) I,IPASS
 1001 FORMAT(I2,A4)
      IF(IPASS.EQ.IPASSO) GOTO 7
      IF(I.EQ.NDA.OR.I.EQ.NF1.OR.I.EQ.NF3) CALL FEHLER('THE PROPOSED UNI
     1T IS RESERVED TO MAPLIB-SYSTEM',46,&1)
    7 GOTO (10,11,12,13,14),K
   10 OUT=I
      GOTO 1
   11 PRT=I
      GOTO 1
   12 INP=I
      GOTO 1
   13 NF1=I
      GOTO 1
   14 NF3=I
      GOTO 1
  999 CALL FEHLER('END OF FILE',11,&51)
  998 CALL FEHLER('ERROR IN READING NEW UNIT-NUMBER',32,&52)
   51 WRITE(6,6) INP,OUT,PRT,NF1,NF3
      RETURN 1
      END
```

```
      SUBROUTINE COPY(WCRT,*,*)
CN    COPY    KOPIEREN VON DATA-SETS
CB    COPY  WIRD NACH DEM KEY COPY AUFGERUFEN.
CB    COPY  ANALYSIERT DIE FOLGE-SCHLUESSELWORTE UND FUEHRT DAS COPIEREN
CB    AUS.
C
CB    STANDARD: .FROM NF3
CB    STANDARD: .TO   PRT
C
C
      COMMON/SKCNT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER WCRT(2),END/'END '/,NPASS/'$COP'/,PASS,A(20),PRT,OUT
     2,ST(2)/'.FRO','.TO '/
      NIN=NF3
      NOUT=PRT
      LOGICAL*1LEND/.FALSE.  /
    1 READ(INP,1000,END=999) WCRT
      IF(WCRT(1).EQ.END) GOTO 1
      CALL INFO(WCRT,I,&100)
      DO 2 I=1,2
      IF(WCRT(1).EQ.ST(I)) GOTO 3
    2 CONTINUE
      CALL FEHLER('KEY WORD MISSING OR WRONG',25,&1)
    3 READ(INP,1001,END=999,ERR=998) J,PASS
 1000 FORMAT(20A4)
 1001 FORMAT(I2,A4)
      GOTO (10,11),I
C   .FROM
   10 NIN=J
      GOTO 1
C   .TO
   11 IF(PASS.EQ.NPASS) GOTO 20
      IF(J.EQ.NF1.OR.J.EQ.NF3.OR.J.EQ.NDA.OR.J.EQ.INP) CALL FEHLER
     1'THIS OUTPUT-UNIT IS NOT ALLOWED',    31,&500)
   20 NOUT=J
      GOTO 1
  100 REWIND NIN
  101 READ(NIN,1000,END=200,ERR=998) A
      WRITE(NOUT,1000) A
      GOTO 101
  200 WRITE(OUT,     2000) NIN,NOUT
 2000 FORMAT('0THE DATA SET ON UNIT ',I2,' HAS BEEN COPIED TO ',I2)
      IF(LEND) RETURN 2
      RETURN 1
  999 LEND=.TRUE.
      GOTO 100
  998 CALL FEHLER('IN COPY WHILE READING',21,&500)
  500 RETURN 1
      END
```

```
      SUBROUTINE PRINTI(NINP,NOUT)
CN    PRINTI
CD    1.5.197C
CB    PRINTI LIEST VON DER EINHEIT NINP KARTEN (80SPALTEN) EIN UND
CB    DRUCKT DIESE AUF DER EINHEIT NOUT (DRUCKER) AUS.
CB    JEWEILS 6C KARTEN ERSCHEINEN AUF 1 SEITE.
CB    AUF JEDER SEITE ERSCHEINT DIE UEBERSCHRIFT NACH FORMAT 99  MIT
CB    FORTLAUFENDER SEITEN-ANGABE
CB    NACH DER LETZTEN KARTE ERSCHEINT DER TEXT NACH FORMAT 102.
CB    DER FILE NINP  WIRD  ZURUECKGESPULT.
      DIMENSION IA(20)
      DATA K/0/,J/1/
    2 READ(NINP,100,END=1)IA
      IF(MOD(K,60).NE.0) GOTO 50
      WRITE(NOUT, 99) J
   99 FORMAT('1MAPLIB-UTILITY',7X,'LIST OF INPUT',T120,'PAGE',I4//)
      J=J+1
   50 CONTINUE
      K=K+1
      WRITE(NOUT,101) K,IA
      GOTO 2
1     WRITE(NOUT,102)
      REWIND NINP
      RETURN
  100 FORMAT(20A4)
  101 FORMAT(' CARD',I5,6X,20A4)
102   FORMAT(/' END OF INPUT'///)
      END
```

```
      SUBROUTINE FILFIL(IN,N,A,LERL)
CN    FILFIL- SCHREIBEN VOM FILE IN AUF DIRECTACCESS
CB    VON DER EINHEIT IN WIRD OHNE VORHERIGES REWIND GELESEN UND AUF
CB    DIRECT-ACCESS GESCHRIEBEN, BIS DAS ERSTE WORT EINES RECORD ='END '
CB    IST. BEI RETURN GIBT IAV DIE ZAHL DER GELESENEN RECORDS PLUS 1 AN.
CB    MIT NEWFIL UND INIT WERDEN AUSGANGSPARAMETER IN GETREC, GETFIL
CB    SOWIE IN FILOUT INITIALISIERT.
CP    IN      LESE-EINHEIT
CP    N       DIMENSION VON A
CP    A       ARBEITSFELD A(20,N)
CP    LERL    INDIKATOR-FELD FUER ERLEDIGTE RECORDS, DASS HIER INITIALI-
CP            SIERT WIRD
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      INTEGER A(20,N),END/'END '/
      LOGICAL*1 LERL(1),NOTEND
      DEFINE FILE 9 (200,1000,U,IAV)
      IAV=1
      K=1
      CALL NEWFIL
      CALL INIT
      NOTEND=.TRUE.
    2 DO1I=1,N
      LERL(K)=.FALSE.
      K=MINO(K+1,4000)
      READ(IN,100,END=999,ERR=3)(A(J,I),J=1,20)
      IF(A(1,I).EQ.END) GOTO 999
    1 CONTINUE
      I=N
  100 FORMAT(20A4)
    5 IF(IAV.GE.NBRREC)CALL FEHLER('NOT ENOUGH SPACE FOR DIRECT ACCESS',
     1   42,&1000)
      WRITE(9'IAV,ERR=4) A
      IF(NOTEND) GOTO2
      IAV=IAV*50-N+I-49
 1000 RETURN
    3 CALL FEHLER('INPUT ERROR IN FILFIL',21,&1000)
    4 CALL FEHLER('ERROR ON DIRECT ACCESS *FILFIL',30,&1000)
  999 NOTEND=.FALSE.
      IF(IAV.GT.1) GOTO 5
      IAV=I+1
      CALL NEW2(I  )
      GOTO 1000
      END
```

```
      SUBROUTINE F11TC9
CN    F11TC9    COPY VON 11 NACH 9
CB    FALLS FUNCDA=.FALSE. WIRD MIT FILFIL DER DATA-SET AUF EINHEIT NF1
CB    AUF NDA COPIERT  UND DANN FUNCDA=.TRUE. GESETZT.
      COMMON/SDIM1/ LSYN(500),LERL(4000),FIL(80,100)
      LOGICAL*1LSYN,LERL,FIL
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      COMMON/SKONT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1   ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,  FUNCDA,NEWMAS,PUNCH
      IF(FUNCDA) RETURN
      REWIND NF1
      CALL FILFIL(NF1,NDIMAX,FIL,LERL)
      FUNCDA=.TRUE.
      RETURN
      END
```

```
      SUBROUTINE FIND11(NRF11,NF1)
CN    FIND11
CB    DIESE ROUTINE SETZT DEN SCHREIB-LESE-KOPF DER EINHEIT NF1
CB    AUF DEN ERSTEN RECORD NACH DEM LETZTEN END-STATEMENT BZW. AUF
CB    DEN RECORD MIT DEM INHALT 'END OF FILE'
CB    DIESE ROUTINE IST SO UNSTAENDLICH PROGRAMMIERT, WEIL MIT DEM
CB    UEBERGANG VOM LESEN ZUM ANSCHLIESSENDEN SCHREIBEN SCHLECHTE
CB    ERFAHRUNGEN GEMACHT WURDEN.
CF    NRF11  ZAHL DER BISHER AUF NF1 GESCHRIEBENEN RECORDS
      INTEGER END/'END '/
      INTEGER M(3),MEND(2)/'  EN','D   '/
      REWIND NF1
      IF(NRF11.EQ.0) GOTO 100
      NRF11=NRF11-2
      DO 776 I=1,NRF11
  776 READ(NF1,1000)K
   11 NRF11=NRF11+1
      READ(NF1,1000)M
 1000 FORMAT(3A4)
      IF(M(2).EQ.MEND(1)  .AND.  M(3).EQ.MEND(2) )GOTO 211
      IF(K.NE.END) GOTO 11
  212 BACKSPACE NF1
      NRF11=NRF11-1
  100 RETURN
  211 READ(NF1,1000) K
      NRF11=NRF11+1
      GOTO 212
C
      ENTRY END11(NF1)
CN    END11
CB    SCHREIBT: 'END OF FILE'
CB    DIESER SATZ WIRD IN FILFIL BENOETIGT (GENAU NUR:'END ')
      WRITE(NF1,1001)
 1001 FORMAT('END OF FILE')
      GOTO 100
      END
```

```
      SUBRCUTINE GETREC(A,IR)
CN    GETREC
CB    GETREC  LIEFERT VCN DIRECT-ACCESS DEN IR-TEN RECORD.
CB    GETREC  LIEST DIE DATEN IN BLCECKEN ZU ZWEIMAL 50 KARTEN IN DAS
CB    ARBEITSFELD IFIL4 (ENTSPRICHT SONST FIL)
      CCMMCN/SDIM1/LSYN4(125),LERL4(1000),IFIL4(2000)
      CCMMCN/SKCNT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,CUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER CUT,PRT
     2   PRT,NRF11,IAV,NREC$$,NRECST,NRECEG,K$,K$O,K$MAX
      INTEGER A(20),FIL(20,100),FIL1(1000),FIL2(1000)
      EQUIVALENCE(IFIL4(1),FIL(1,1),FIL1(1)),(IFIL4(1001),FIL2(1))
      IF(I1.GT.IR.CR.I2.LT.IR) GCTO 10
  100 I=IR-I1+1
      DC 1 J=1,2C
    1 A(J)=FIL(J,I)
   2C RETURN
   10 J1=(IR-1)/50+1
      IF(J1.LT.IRMIP2)J1=IRMIN
      IF(J1.GT.NBRREC) CALL FEHLER('J1.GT.NBRREC *GETREC',28,&30)
      J2=MINO(J1+1,NBRREC)
      I1=(J1-1)*5C+1
      I2=J2*50
C
CB    MIT I1 UNC I2 WIRC NCTIERT WELCHE RECORDS EINGELESEN SINC
C
      IF(J1.NE.J2C) GCTC 1C1
      DC 2CC J=1,1CCC
  2CC FIL1(J)=FIL2(J)
      J1C=J1
      J2C=J2
      IAV=J2
      READ(9'IAV) FIL2
      GCTC 100
  1C1 IF(J1.NE.J1C-1) GCTC 1C2
      DC 2C1 J=1,1000
  201 FIL2(J)=FIL1(J)
      J1C=J1
      J2C=J2
      IAV=J1
      READ(9'IAV) FIL1
      GCTC 100
  102 IAV=J1
      READ(9'IAV) FIL1
      IAV=J2
      READ(9'IAV) FIL2
      J1C=J1
      J2C=J2
      GCTC 100
   3C STCP
C
      ENTRY CREATE
CN    CREATE
CB    DEFINE FILE
C
      DEFINE FILE 9 (200,1CCC,U,IAV)
```

```
      GOTO 20
C
      ENTRY FROMRC(IRMI)
CN    FROMRC
CP    IRMI  GIBT AN, WELCHES DIE KLEINSTE NOCH BENOETIGTE RECORD-NR. IST
      IRMIN=(IRMI-1)/50+1
      IRMIP2=IRMIN+2
      GOTO 20
C
      ENTRY NEWFIL
CN    NEWFIL
CB    INITIALISIERUNG
      I1=0
      I2=0
      IRMIN=1
      IRMIP2=3
      J10=-5
      J20=NEW$$$(J10)
      GOTO 20
      ENTRY NEW2(I)
CN    NEW2
CB    ENTRY FUER DEN FALL, DASS MAXIMAL 100 KARTEN EINGELESEN WURDEN
      I1=1
      I2=I
      J10=1
      J20=2
      GOTO 20
      END
```

```
      FUNCTION GETFIL(II,IR)
CN    GETFIL
CB    GETFIL  LIEFERT VON DIRECT-ACCESS DAS II-TE BYTE IM IR-TEN RECORD
CB    GETFIL  HOLT DIE DATEN RECORD-WEISE MIT GETREC
      LOGICAL*1 GETFIL,FIL(80)
      INTEGER A(20)
      EQUIVALENCE(FIL(1),A(1))
      IF(IR.EQ.K) GOTO 1
      CALL GETREC(A,IR)
      K=IR
CB    IN K WIRD NOTIERT, WELCHER RECORD ZULETZT GELESEN WURDE
    1 GETFIL=FIL(II)
   10 RETURN
      ENTRY NEW$$$(I)
CN    NEW$$$
CB    DUMMY-ENTRY ZUR INITIALISIERUNG VON K
      K=0
      NEW$$$=I
      GOTO 10
      END
```

```
      SUBROUTINE KATCON(KAT2,KAT4,A2,A4,JO,J,JMAX,*)
CN    KATCON - KATALOG-CONTROLLE
CC    1.2.70
CB    GESUCHT WIRD  DAS ELEMENT A2,A4, DAS AN ENTSPRECHENDEN POSITIONEN
CB    IN  DEN FELDERN KAT2,KAT4 VORHANDEN SEIN SOLL
CB    GESUCHT WIRD VOM JO-TEN BIS JMAX-TEN ELEMENT
CB    FALLS A2,A4 IN KAT2,KAT4 EXISTIERT,GIBT J SEINE POSITION AN,
CB    ANDERNFALLS WIRD UEBER * ZURUECKGESPRUNGEN
CB    VERGLEICHE LSTCON
      INTEGER*2 A2,KAT2(JMAX)
      INTEGER   A4,KAT4(JMAX)
      IF(JO.GT.JMAX)RETURN 1
      DO 1 J=JO,JMAX
      IF(KAT2(J).NE.A2) GOTO 1
      IF(KAT4(J).EQ.A4) RETURN
    1 CONTINUE
      RETURN1
      END
```

```
      SUBROUTINE LSTCCN(ARG,ARGKAT,IA,I1,I,KA,*)
CN    LSTCCN  -  LISTEN-CONTROLLE
CC    1.5.7C
CB    GESUCHT WIRD IN  INTEGER*2ARGKAT(IA,KA) NACH INTEGER*2 ARG(IA)
CB    IN DEM BEREICH VCN ELEMENT I1 BIS KA
CB    FALLS VORHANDEN GIBT I DIE POSITION AN
CB    ANDERNFALLS RUECKSPRUNG UEBER *
CB    MIT DIESER ROUTINE  KANN AUCH NACH  REAL*4ARG(1)=INTEGER*2(2)
CB    SCWIE NACH REAL*8ARG(1)=INTEBER*2(4) GESUCHT WERDEN
      INTEGER*2 ARG(IA),ARGKAT(IA,KA)
      IF(KA.EQ.0) GOTC 3
      DC 1 I=I1,KA
      DC 2 J=1,IA
      IF(ARG(J).NE.ARGKAT(J,I))GOTO 1
    2 CONTINUE
      RETURN
    1 CONTINUE
    3 RETURN 1
      END
```

```
      SUBROUTINE SUCHR(TEXT,IT,C1,R1,R2,IR,*)
CN    SUCHR - SUCHEN EINES RECORDS MIT DEM SPEZIFIZIERTEN TEXT
CC    1.2.7C
CB    DIESE ROUTINE SUCHT DEN TEXT IN EINEM  DATA-SET.
CB    DER TEXT  SOLL  IN  SPALTE C1  BEGINNEN,
CB    ER SOLL IT ZEICHEN LANG SEIN
CB    ER SOLL ZWISCHEN DER  R1-TEN  UND DER R2-TEN KARTE STEHEN (JEWEILS
CB    EINSCHLIESSLICH)
CB    IST DER TEXT VORHANDEN, SO GIBT IR DIE KARTEN-NUMMER AN, DIE DIE
CB    KARTE MIT DEM TEXT KENNZEICHNET
CB    WIRD DER TEXT NICHT GEFUNDEN, SO WIRD UEBER * ZURUECKGESPRUNGEN
      LOGICAL*1 TEXT(IT),TE(2,80),TA(2,80),GETFIL
      INTEGER*2 E(8C)/80*'  '/,A(8C)/80*'  '/
      EQUIVALENCE (E(1),TE(1,1)),(A(1),TA(1,1))
      INTEGER C1,C2,R1,R2
      DC 1 I=1,IT
    1 TE(1,C1+I-1)=TEXT(I)
      C2=C1+IT-1
      DO 2 IA=R1,R2
      DC 3 I=C1,C2
      TA(1,I)=GETFIL(I,IA)
      IF(A(I).NE.E(I)) GCTC2
    3 CCNTINUE
      IR=IA
      RETURN
    2 CCNTINUE
      RETURN 1
      END
```

```
      SUBROUTINE SUCHC(Q,Q1,I1,I2,I,*)
CN    SUCHC - SUCHEN DER SPALTE(COLUMN) MIT DEM ZEICHEN Q1
CC    1.2.7C
CB    DAS FELD Q(8C) WIRD VCN SPALTE I1 BIS I2 NACH DEM ELEMENT Q1
CB    DURCHSUCHT
CB    FALLS Q1 VCRHANDEN IST, GIBT I DIE PCSITION AN
CB    ANDERNFALLS WIRD UEBER * ZURUECKGESPRUNGEN
CB    IM ALLGEMEINEN WIRD SUCHC ZUM SUCHEN EINES EINZELNEN ZEICHENS
CB    GESUCHT; ES MUSS HIERZU ERST VON LOGICAL*1 NACH INTEGER*2 UMGE-
CB    SPEICHERT WERDEN
      INTEGER*2Q(8C),Q1
      DC 1 I=I1,I2
      IF(Q(I).EQ.Q1) RETURN
    1 CCNTINUE
      RETURN1
      END
```

```
      FUNCTION NRSYN(J)
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTOA(100),NRSTOE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
      COMMON/SKONT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
CN    NRSYN - NUMMER DES SYNONYMS
CB    NRSYN LIEFERT DIE SYNONYM-NR IM SYNONYM-KATALOG ZUR FUNCTION NR J
CB    I=NRSYN(J) IST DIE UMKEHRFUNKTION ZU J=JSYN(I)
CB    VORAUSGESETZT WIRD, DASS ZUR FUNCTION NR.J EIN SYNONYM EXISTIERT
      DO 1 I=1,KY
      IF(JSYN(I).EQ.J) GOTO 2
    1 CONTINUE
      CALL FEHLER('IN NRSYN',9,82)
    2 NRSYN=I
      RETURN
      END
```

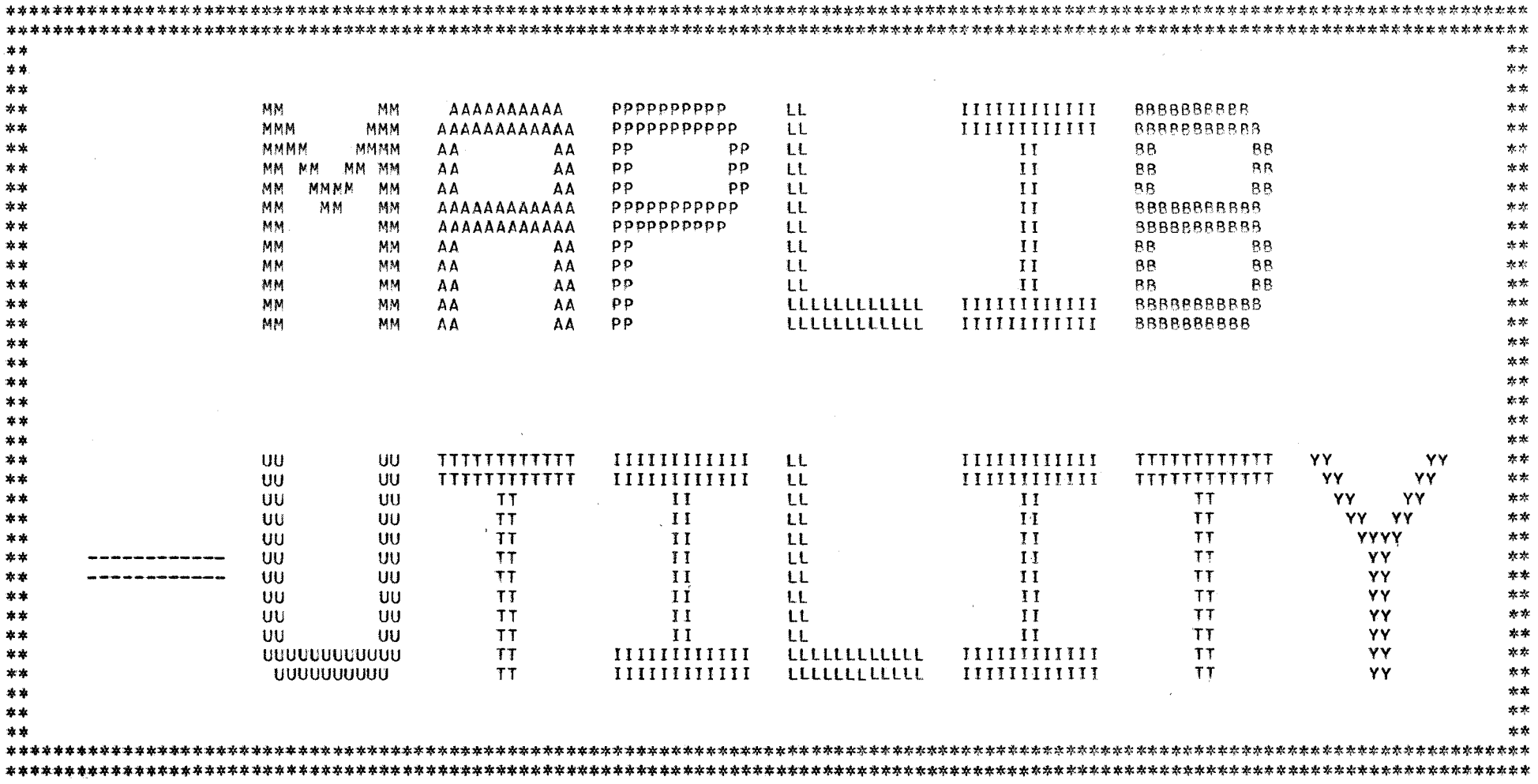
```
      SUBROUTINE FEHLER(TEXT,I,*)
      COMMON/SKCNT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1   ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,  FUNCDA,NEWMAS,PUNCH
CN    FEHLER
CB    NACH FORMAT 100 WIRD DER AUS I ZEICHEN BESTEHENDE TEXT AUSGEDRUCKT
CB    (STETS AUF EINHEIT 6),
CB    DER INDIKATOR FUER EINGABEFEHLER FERROR WIRD TRUE GESETZT
CB    ES WIRD STETS UEBER * ZURUECKGESPRUNGEN
CD    1.2.70
C
      LOGICAL*1 TEXT(I)
      FERROR =.TRUE.
      WRITE(6,100) TEXT
  100 FORMAT('0*** ERROR ***     ',(100A1))
      RETURN 1
      END
```

```
      SUBROUTINE PUBLIC(TEXT)
CN    PUBLIC -  FUER PUBLIC-RELATIONS
CD    1.4.7C
CB    DIENT ZUM AUSDRUCKEN DER IN TEXT ENTHALTENEN 8 ZEICHEN IN GROSS-
CB    FORMAT, SOWIE ZUR AUSGABE VON STANDARDINFORMATION:
CB    DAT =DATUM
CB    UHR =UHRZEIT
CB    JEIN=ZAHL DER INTEGRIERTEN ROUTINEN
CB    KS  =ZAHL DER INTEGRIERTEN STOFFE
CB    KE  =ZAHL DER INTEGRIERTEN EIGENSCHAFTEN
CB    I   =FREIER KERNSPEICHER
CS    DATUM UND FREESP SIND SYSTEM-ROUTINEN
CB    DIE AUSGABE ERFOLGT NACH SEITENVORSCHUB IMMER AUF EINHEIT 6
CE    UNABHAENGIG VON OUT
CE    PUBLIC WIRD NACH JEDEM HAUPTSCHLUESSELWORT UND VOR STOP AUFGERUFEN
      COMMON/SKCNT8/RELDAT,RELUHR,DAT,UHR
      REAL*8 RELDAT,RELUHR,DAT,UHR
      COMMON/SKCNT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      REAL*8TEXT
      WRITE(6  ,1)
1     FORMAT('1')
      CALL A8FORM(6  ,TEXT)
      CALL DATUM(DAT,UHR)
      WRITE(6  ,11)DAT,UHR,JEIN,KS,KE
   11 FORMAT(//////'0',T50,'DATE' ,2X,A8,5X,'TIME',2X,A8////
     1'0',T50,'IN  MAPLIB  ARE  INTEGRATED'/
     2'0',T55,I5,'  FUNCTIONS'/
     3'0',T55,I5,'  MATERIALS'/
     4'0',T55,I5,'  PROPERTIES'/)
      WRITE(6,222) RELDAT,RELUHR
  222 FORMAT('0',T43,'LATEST CHANGE OF MAPLIB: ',A8,' AT ',A8)
      CALL FREESP(I)
      WRITE (6  ,1020) I
 1020 FORMAT('0'/'0'/'0'/'0','FREE REGION :',I4,' K BYTES')
      RETURN
      END
```

```
      BLOCK DATA
CB    INITIALISIERUNG DER COMMON-FELDER
      COMMON/SKCNT8/RELDAT,RELUHR,DAT,UHR
      REAL*8 RELDAT,RELUHR,DAT,UHR
      COMMON/SKCNT4/ NDIMAX,NBRREC,NDA,NRLEN,JEINO,JEIN,JEMAX,KE,KEO,
     1KEMAX,KS,KSO,KSMAX,KA,KAO,KAMAX,KY,KYO,KYMAX,INP,OUT,NF1,NF2,NF3,
     2 PRT,NRF11,IAV,N$$A,N$$E,K$,K$O,K$MAX
      INTEGER OUT,PRT
      COMMON/SKCNT2/ NAMAX,NAMAXO
      INTEGER*2 NAMAX,NAMAXO
      COMMON/SKCNT1/ ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,
     1    ERKFOL,FUNCDA,NEWMAS,PUNCH
      LOGICAL*1 ERROR,FERROR,NURSOR,NOTARG,NOTEIG,NOTSTO,NOTDOU,ERKFOL
     1,   FUNCDA,NEWMAS,PUNCH
      COMMON/SDIM4/ STO( 500),ARG(6,500),NRECA(500),NRECE(500),KATSTO
     1(100),SYNSTO(20),JSYN(20),ERKEIG(17,100),ERKSTO(17,100),ERKARG
     2(17,50),NRSTOA(100),NRSTOE(100),NREIGA(100),NREIGE(100),
     3NRECSA( 50),NRECSE(50 )
      INTEGER STO,ARG,SYNSTO,ERKEIG,ERKSTO,ERKARG
C
      DATA RELDAT/'  NEW    '/,RELUHR/'  NOW    '/
      DATA  NDIMAX  /100/
     2,NBRREC/200/
     3,NDA    /9/
     4,NRLEN /80/
     5,JEIN  /0/
     6,JEMAX /500/
     7,KE    /0/
     8,KEO   /0/
     9,KEMAX /100/
     1,KS/0/
     1,KSO   /0/
     2,KSMAX /100/
     3,KA    /0/
     4,KAO   /0/
     5,KAMAX /50/
     6,KY    /0/
     7,KYO   /0/
     8,KYMAX /20/
     9,JEINO /0/
      DATA INP/5/
     2, OUT   /6/
     3, NF1   /11/
     4,NF2   /11/
     5, NF3   /13/
     6, PRT   /7/
     7, NRF11 /0/
     8, N$$A  /0/
     9, N$$E  /0/
     1,K$  /0/
     1, K$O   /0/
     2, K$MAX /50/
      DATA NAMAX/0/
     2,NAMAXO/0/
      DATA ERROR/.FALSE./
     2, FERROR /.FALSE./
     3, NURSOR /.FALSE./
```

A n h a n g 8

B e i s p i e l a u s g a b e



RELEASE DATE OF MAPLIB-UTILITY: 12.07.70

```
CARD     1       OLD             BEZUG AUF BISHERIGEN BIBLIOTHEKSBESTAND
CARD     2       INFORMATION     AUSGABE VON INFORMATION
CARD     3       .MAP            UEBERSICHT UBER DIE INTEGRIERTEN ROUTINEN
CARD     4       ..$$$$$$        MASTERFUNKTION DER OBERSTEN EBENE
CARD     5       ..$$HEV         MASTERFUNKTION DER STOFFEBENE
CARD     6       ..ES$$$$        MASTERFUNKTION DER EIGENSCHAFTSEBENE
CARD     7       ..ESHEV         DATEN-FUNKTION

END OF INPUT
```

```
  OOOOOOOOO     LL              DDDDDDDDDD
 OOOOOOOOOOO    LL              DDDDDDDDDDD
OO         OO   LL              DD         DD
OO         OO   LL              DD         DD
OO         OO   LL              DD         DD
OO         OO   LL              DD         DD
OO         OO   LL              DD         DD
OO         OO   LL              DD         DD
OO         OO   LL              DD         DD
OO         OO   LL              DD         DD
 OOOOOOOOOOO    LLLLLLLLLLLL    DDDDDDDDDDD
  OOOOOOOOO     LLLLLLLLLLLL    DDDDDDDDDD
```

DATE 04.08.70 TIME 04.07.14

IN MAPLIB ARE INTEGRATED

121 FUNCTIONS

14 MATERIALS

23 PROPERTIES

LATEST CHANGE OF MAPLIB: 24.07.70 AT 01.32.50

FREE REGION : 44 K BYTES

```
IIIIIIIIIIII    NN          NN    FFFFFFFFFFFF     0000000000
IIIIIIIIIIII    NNN         NN    FFFFFFFFFFFF    000000000000
     II         NNNN        NN    FF              00        00
     II         NN NN       NN    FF              00        00
     II         NN  NN      NN    FF              00        00
     II         NN   NN     NN    FFFFFFFF        00        00
     II         NN    NN    NN    FFFFFFFF        00        00
     II         NN     NN   NN    FF              00        00
     II         NN      NNNN    FF              00        00
     II         NN       NNN    FF              00        00
IIIIIIIIIIII    NN         NN    FF              000000000000
IIIIIIIIIIII    NN          N    FF               0000000000
```

DATE 04.08.70 TIME 04.07.15

IN MAPLIB ARE INTEGRATED

121 FUNCTIONS

14 MATERIALS

23 PROPERTIES

LATEST CHANGE OF MAPLIB: 24.07.70 AT 01.32.50

FREE REGION : 44 K BYTES

SYMBOL	MATERIAL	
4961	STAHL 4961	
4981	STAHL 4981	
4988	STAHL 4988	
HEV	HELIUM-GAS	
B4C	BOR-4-CARBID	(B4C)
UO	URAN OXID	(UO2)
PUO	PLUTONIUM OXID	(PUO2)
UPUO	URAN-PLUTONIUM MISCHOXID	
NA	NATRIUM	
NAV	NATRIUM-DAMPF	
NAL	NATRIUM, FLUESSIG	
NALS	GESAETTIGTES FLUESSIGES NATRIUM	
NAVS	GESAETTIGTER NATRIUMDAMPF	
CO2V	CO2-GAS	

** PROPERTIES

SYMBOL	PROPERTY	UNIT
FT	SCHMELZTEMPERATUR	K
RO	DICHTE	KG/M3
GA	LINEARER,DIFF. AUSDEHNUNGSKOEFFIZIENT	1/K
EH	ENTHALPIE	J/KG
ES	ENTROPIE	J/KG.K
ZD	DYNAMISCHE VISKOSITAET	N.S/M2
WL	WAERMELEITFAEHIGKEIT	W/M.K
CP	SPEZ. WAERMEKAPAZITAET B. CONST. DRUCK	J/KG.K
CV	SPEZ. WAERMEKAPAZITAET B. CONST. VOLUMEN	J/KG.K
PR	PRANDTL-ZAHL	1
RS	SPEZIELLE GASKONSTANTE	J/KG.K
FH	SCHMELZWAERME	J/KG
RH	REKRISTALLISATIONSWAERME	J/KG
EM	ELASTIZITAETSMODUL	N/M2
VO	SPEZIFISCHES VOLUMEN	M3/KG
VP	SATTDAMPFDRUCK	N/M2
VT	SATTDAMPFTEMPERATUR	K
VH	VERDAMPFUNGSENTHALPIE	J/KG
VS	VERDAMPFUNGSENTROPIE	J/KG.K
PV	DRUCK ALS FUNKTION VON TEMPERATUR UND VOLUMEN	N/M2
ST	OBERFLAECHEN-SPANNUNG	N/M
SD	DEHNGRENZE	N/M2
SB	BRUCHSPANNUNG	N/M2

** PARAMETERS

SYMBOL	PARAMETER	UNIT
I	DIESER PARAMETER-NAME IST VERBOTEN	-
TK	TEMPERATUR	K
DUMMY	DUMMY-PARAMETER	
PN	DRUCK	N/M2
CNPU	KONZENTRATION DES PLUTONIUMOXIDS	MOL-BRUCH
PORVOL	RELATIVES PORVOLUMEN	VOL-ANTEIL
XPOR	RELATIVES PORVOLUMEN; XPOR= PORVOL	VOL-ANTEIL
PNM2	DRUCK	N/M2
V	SPEZIFISCHES VOLUMEN	M3/KG
T	TEMPERATUR	K
STANDT	STANDZEIT	S
PERCEN	DEHNUNGSKOEFFIZIENT	%

*** TABLE OF THE CATALOGUED FUNCTIONS
MARKED BY THEIR PARAMETER-LIST

PROPERTY	/ MATERIAL 4961	4981	4988	HEV	B4C
FT	(DUMMY)	(DUMMY)	(DUMMY)	*	(DUMMY)
RO	(TK)	(TK)	(TK)	*	(TK)
GA	(TK)	(TK)	(TK)	*	(TK)
EH	*	*	*	(TK,PN)	*
ES	*	*	*	(TK,PN)	*
ZD	*	*	*	(TK,PN)	*
WL	(TK)	(TK)	(TK)	(TK,PN)	(TK)
CP	(TK)	(TK)	(TK)	(TK,PN)	(TK)
CV	*	*	*	(TK,PN)	*
PR	*	*	*	(TK,PN)	*
RS	*	*	*	(DUMMY)	*
FH	*	*	*	*	*
RH	*	*	*	*	*
EM	(TK)	(TK)	(TK)	*	*
VC	*	*	*	(TK,PN)	*
VP	*	*	*	*	*
VT	*	*	*	*	*
VH	*	*	*	*	*
VS	*	*	*	*	*
PV	*	*	*	*	*
ST	*	*	*	*	*
SD	*	(T,STANDT,PERCEN)	*	*	*
SB	*	(T,STANDT)	*	*	*

MATERIALS CONTINUED

PROPERTY	/ MATERIAL UO	PUO	UPUO	NA	NAV
FT	(DUMMY)	(DUMMY)	(CNPU)	*	*
RO	(TK;PORVOL)	(TK;PORVOL)	(TK,CNPU;PORVOL)	*	(TK,PNM2)
GA	(TK)	(TK)	(TK)	*	*
EH	*	*	*	*	(TK,PNM2)
ES	*	*	*	*	(TK,PNM2)
ZD	*	*	*	*	(TK)
WL	(TK,XPOR)	(TK,XPOR)	(TK,XPOR)	*	(TK)
CP	(TK)	(TK)	(TK)	*	*
CV	*	*	*	*	*
PR	*	*	*	*	(TK)
RS	*	*	*	*	*
FH	(DUMMY)	(DUMMY)	(DUMMY)	*	*
RH	(DUMMY)	(DUMMY)	(DUMMY)	*	*
EM	*	*	*	*	*
VO	*	*	*	*	(TK,PNM2)
VP	*	*	*	(TK)	(TK)
VT	*	*	*	(PNM2)	(PNM2)
VH	*	*	*	(TK)	(TK)
VS	*	*	*	(TK)	*
PV	*	*	*	*	(TK,V)
ST	*	*	*	(TK)	*
SD	*	*	*	*	*
SB	*	*	*	*	*

MATERIALS CONTINUED

PROPERTY / MATERIAL	NAL	NALS	NAVS	CO2V
FT	*	*	*	*
RO	(TK)	(TK)	(TK)	(TK,PN)
GA	*	*	*	*
EH	(TK,PNM2)	(TK)	(TK)	(TK,PN)
ES	(TK,PNM2)	(TK)	(TK)	(TK,PN)
ZD	(TK)	(TK)	(TK)	(TK,PN)
WL	(TK)	(TK)	(TK)	(TK,PN)
CP	*	(TK)	(TK)	(TK,PN)
CV	*	*	*	(TK,PN)
PR	(TK)	(TK)	(TK)	(TK,PN)
RS	*	*	*	(DUMMY)
FH	*	*	*	*
RH	*	*	*	*
EM	*	*	*	*
VO	(TK)	(TK)	(TK)	(TK,PN)
VP	(TK)	(TK)	(TK)	(TK)
VT	(PNM2)	(PNM2)	(PNM2)	(PN)
VH	(TK)	(TK)	(TK)	*
VS	(TK)	(TK)	(TK)	*
PV	*	*	(TK)	*
ST	(TK)	(TK)	*	*
SD	*	*	*	*
SB	*	*	*	*

* EXPLANATIONS

IF THERE EXISTS A PARAMETER-LIST, THE FUNCTION DEFINED BY THE COMBINATION OF THE PROPERTY- AND MATERIAL-SYMBOL IS INTEGRATED IN MAPLIB AND CAN BE CALLED WITH THE PARAMETERS SHOWN.

IF THERE IS ONLY A '*', THE FUNCTION IS NOT INTEGRATED IN MAPLIB; PLEASE ADD.

IF THE PARAMETER-LIST CONSISTS OUT OF A LOT OF PARAMETER-SYMBOLS WITH ONLY ONE CHARACTER OR DOES NOT END WITH A PARENTHESIS, THE PARAMETER-SYMBOLS ARE ABBREVIATED; REFER TO THE FUNCTION-LIST FOR THE COMPLETE PARAMETER-LIST.

IF IN THE PARAMETER-LIST A ',' IS SUBSTITUTED BY A ';', THE PARAMETERS AFTER THE ';' ARE OPTIONAL.

** NO SYNONYM-NAMES ARE DEFINED

```
**  LIST OF INTEGRATED SYSTEM ROUTINES

    1   STAT$
    2   $WARN
    3   $NUMBR
    4   ERR$$$
    5   $CNT
    6   $STAT7
    7   $ERROR
    8   RAGVL1
    9   RANGE
   10   $RANGE
   11   $WARN2
   12   RANGE1
   13   RANGE2
   14   X1X2X4
   15   $NEWS
   16   TEST$
   17   $TEST
   18   VERCO2

** NUMBER OF RECORDS ON THE SOURCE FILE:     5312
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$       FUNCTION $$$$$$(/EIG/,/STOFF/,/A1/,/A2/,/A3/)                   $$$$0001$
$C                                                                      $$$$0002$
$CN     ***  M A P L I B  ***    RETRIEVAL  FUNCTION                    $$$$0003$
$C                                                                      $$$$0004$
$CA     MAPLIB-UTILITY                                                  $$$$0005$
$CD     15.07.70                                                        $$$$0006$
$       COMMON /$NUMB$/ J,LJ                                            $$$$0007$
$       LOGICAL*1 LJ                                                    $$$$0008$
$       INTEGER*2 EIG                                                   $$$$0009$
$       INTEGER STOFF,STOKAT( 14)/'4961','4981','4988','HEV ','B4C ',   $$$$0010$
$      1'UO  ','PUO ','UPUO','NA  ','NAV ','NAL ','NALS','NAVS','CO2V'/ $$$$0011$
$      1                                                                $$$$0012$
$       J=NUMBR$( 3)-2                                                  $$$$0013$
$       IF (J.LE.-1) GOTO 9996                                          $$$$0014$
$       LJ=.FALSE.                                                      $$$$0015$
$       DO 9998 I=1, 14                                                 $$$$0016$
$       IF (STOFF.EQ.STOKAT(I)) GOTO ( 1, 2, 3, 4, 5, 6, 7, 8, 9,10,11, $$$$0017$
$      112,13,14)                                                       $$$$0018$
$      1 ,I                                                             $$$$0019$
$ 9998 CONTINUE                                                         $$$$0020$
$ 9996 $$$$$$=ERR$$$(STOFF)                                             $$$$0021$
$ 9999 RETURN                                                           $$$$0022$
$    1 $$$$$$=$$4961(EIG,A 1)                                           $$$$0023$
$      GO TO 9999                                                       $$$$0024$
$    2 $$$$$$=$$4981(EIG,A 1,A 2,A 3)                                   $$$$0025$
$      GO TO 9999                                                       $$$$0026$
$    3 $$$$$$=$$4988(EIG,A 1)                                           $$$$0027$
$      GO TO 9999                                                       $$$$0028$
$    4 $$$$$$=$$HEV (EIG,A 1,A 2)                                       $$$$0029$
$      GO TO 9999                                                       $$$$0030$
$    5 $$$$$$=$$B4C (EIG,A 1)                                           $$$$0031$
$      GO TO 9999                                                       $$$$0032$
$    6 $$$$$$=$$UO  (EIG,A 1,A 2)                                       $$$$0033$
$      GO TO 9999                                                       $$$$0034$
$    7 $$$$$$=$$PUO (EIG,A 1,A 2)                                       $$$$0035$
$      GO TO 9999                                                       $$$$0036$
$    8 $$$$$$=$$UPUO(EIG,A 1,A 2,A 3)                                   $$$$0037$
$      GO TO 9999                                                       $$$$0038$
$    9 $$$$$$=$$NA  (EIG,A 1)                                           $$$$0039$
$      GO TO 9999                                                       $$$$0040$
$   10 $$$$$$=$$NAV (EIG,A 1,A 2)                                       $$$$0041$
$      GO TO 9999                                                       $$$$0042$
$   11 $$$$$$=$$NAL (EIG,A 1,A 2)                                       $$$$0043$
$      GO TO 9999                                                       $$$$0044$
$   12 $$$$$$=$$NALS(EIG,A 1)                                           $$$$0045$
$      GO TO 9999                                                       $$$$0046$
$   13 $$$$$$=$$NAVS(EIG,A 1)                                           $$$$0047$
$      GO TO 9999                                                       $$$$0048$
$   14 $$$$$$=$$CO2V(EIG,A 1,A 2)                                       $$$$0049$
$      GO TO 9999                                                       $$$$0050$
$      END                                                              $$$$0051$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      FUNCTION $$HEV (/EIG/,/A1/,/A2/)                                       $$HE0052$
$C                                                                            $$HE0053$
$CN    ***  M A P L I B  ***    RETRIEVAL  FUNCTION                            $$HE0054$
$C                                                                            $$HE0055$
$CA    MAPLIB-UTILITY                                                          $$HE0056$
$CD    15.07.70                                                                $$HE0057$
$C$M   HELIUM-GAS                                                              $$HE0058$
$      COMMON /$NUMB$/ J,LJ                                                    $$HE0059$
$      LOGICAL*1 LJ                                                            $$HE0060$
$      INTEGER*2 EIG,VALUE(4)/'HEV     '/,EIGKAT(  9)/'EH','ES','ZD',          $$HE0061$
$     1'WL','CP','CV','PR','RS','VO'/                                          $$HE0062$
$      IF (LJ) J=NUMBR$( 2)-1                                                  $$HE0063$
$      LJ=.FALSE.                                                              $$HE0064$
$      DO 9998 I=1,  9                                                         $$HE0065$
$      IF (EIG.EQ.EIGKAT(I)) GOTO(1, 2, 3, 4, 5, 6, 7, 8, 9)                   $$HE0066$
$     1 ,I                                                                     $$HE0067$
$ 9998 CONTINUE                                                                $$HE0068$
$      VALUE(3)=EIG                                                            $$HE0069$
$      $$HEV =ERR$X$(VALUE)                                                    $$HE0070$
$ 9999 LJ=.TRUE.                                                               $$HE0071$
$      RETURN                                                                  $$HE0072$
$    1 $$HEV =EHHEV (A1,A 2)                                                   $$HE0073$
$      GO TO 9999                                                              $$HE0074$
$    2 $$HEV =ESHEV (A1,A 2)                                                   $$HE0075$
$      GO TO 9999                                                              $$HE0076$
$    3 $$HEV =ZDHEV (A1,A 2)                                                   $$HE0077$
$      GO TO 9999                                                              $$HE0078$
$    4 $$HEV =WLHEV (A1,A 2)                                                   $$HE0079$
$      GO TO 9999                                                              $$HE0080$
$    5 $$HEV =CPHEV (A1,A 2)                                                   $$HE0081$
$      GO TO 9999                                                              $$HE0082$
$    6 $$HEV =CVHEV (A1,A 2)                                                   $$HE0083$
$      GO TO 9999                                                              $$HE0084$
$    7 $$HEV =PRHEV (A1,A 2)                                                   $$HE0085$
$      GO TO 9999                                                              $$HE0086$
$    8 $$HEV =RSHEV (A1)                                                       $$HE0087$
$      GO TO 9999                                                              $$HE0088$
$    9 $$HEV =VOHEV (A1,A 2)                                                   $$HE0089$
$      GO TO 9999                                                              $$HE0090$
$      END                                                                     $$HE0091$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
              $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
CARD    1     $       FUNCTION ES$$$$(/STOFF/,/A1/,/A2/)                              ES$$0092$
CARD    2     $C                                                                      ES$$0093$
CARD    3     $CN     ***  M A P L I B  ***    RETRIEVAL  FUNCTION                    ES$$0094$
CARD    4     $C                                                                      ES$$0095$
CARD    5     $CA     MAPLIB-UTILITY                                                  ES$$0096$
CARD    6     $CD     15.07.70                                                        ES$$0097$
CARD    7     $C$P    ENTROPIE                                           J/KG*GRAD K ES$$0098$
CARD    8     $       COMMON /$NUMB$/ J,LJ                                            ES$$0099$
CARD    9     $       LOGICAL*1 LJ                                                    ES$$0100$
CARD   10     $       INTEGER VALUE(2)/'ES    '/,STOFF,STOKAT(  6)/'HEV ','NALS',     ES$$0101$
CARD   11     $      1'NAV ','NAVS','NAL ','CO2V'/                                    ES$$0102$
CARD   12     $       IF (LJ) J=NUMBR$( 2)-1                                          ES$$0103$
CARD   13     $       LJ=.FALSE.                                                      ES$$0104$
CARD   14     $       DO 9998 I=1,  6                                                 ES$$0105$
CARD   15     $       IF (STOFF.EQ.STOKAT(I)) GOTO(1, 2, 3, 4, 5, 6)                  ES$$0106$
CARD   16     $      1 ,I                                                             ES$$0107$
CARD   17     $ 9998 CONTINUE                                                         ES$$0108$
CARD   18     $       VALUE(2)=STOFF                                                  ES$$0109$
CARD   19     $       ES$$$$=ERRX$$(VALUE)                                            ES$$0110$
CARD   20     $ 9999 LJ=.TRUE.                                                        ES$$0111$
CARD   21     $       RETURN                                                          ES$$0112$
CARD   22     $     1 ES$$$$=ESHEV (A1,A 2)                                           ES$$0113$
CARD   23     $       GO TO 9999                                                      ES$$0114$
CARD   24     $     2 ES$$$$=ESNALS(A1)                                               ES$$0115$
CARD   25     $       GO TO 9999                                                      ES$$0116$
CARD   26     $     3 ES$$$$=ESNAV (A1,A 2)                                           ES$$0117$
CARD   27     $       GO TO 9999                                                      ES$$0118$
CARD   28     $     4 ES$$$$=ESNAVS(A1)                                               ES$$0119$
CARD   29     $       GO TO 9999                                                      ES$$0120$
CARD   30     $     5 ES$$$$=ESNAL (A1,A 2)                                           ES$$0121$
CARD   31     $       GO TO 9999                                                      ES$$0122$
CARD   32     $     6 ES$$$$=ESCO2V(A1,A 2)                                           ES$$0123$
CARD   33     $       GO TO 9999                                                      ES$$0124$
CARD   34     $       END                                                             ES$$0125$
              $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$       FUNCTION ESHEV (/TK     /,/PN     /)                             ESHE0126$
$C                                                                       ESHE0127$
$CN     ***  M A P L I B  ***    FUNCTION                                ESHE0128$
$C                                                                       ESHE0129$
$CD     22.07.70                                                         ESHE0130$
$CA     SCHNAUDER & ZIMMERER                                             ESHE0131$
$C$P    ENTROPIE                                          J/KG.K         ESHE0132$
$C$M    HELIUM-GAS                                                       ESHE0133$
$CP     TK     10.<=273.<=TK<=1473.<=50000.                              ESHE0134$
$CP     TEMPERATUR                                        K              ESHE0135$
$CP     PN     1.<=9.8067E4<=PN<=9.8067E6<=9.8067E7                      ESHE0136$
$CP     DRUCK                                             N/M2           ESHE0137$
$CL     K. BAMMERT,DIE THERMODYNAMISCHEN EIGENSCHAFTEN VON HELIUM ALS    ESHE0138$
$CL     ARBEITSMEDIUM FUER NUKLEARE GASTURBINEN                          ESHE0139$
$CL     KERNTECHNIK,11(1969),88-91                                       ESHE0140$
$       COMMON/$TEST$/ NOTEST                                            ESHE0141$
$       LOGICAL*1 NOTEST                                                 ESHE0142$
$C$F                                                                     ESHE0143$
$       IF(NOTEST) GOTO 99999                                            ESHE0144$
$       CALL $NUMBR(ESHEV ,'ESHEV ',NUMBR$( 2), 2,1,I,&  999)            ESHE0145$
$       CALL RANGE(ESHEV,'ESHEV ',1,273.,1473.,10.,50000.,TK,&999)       ESHE0146$
$       CALL RANGE(ESHEV,'ESHEV ',2,9.8067E4,9.8067E6,1.,9.8 6 E7,PN,&999)ESHE0147$
$C$T                                                                     ESHE0148$
$99999 ESHEV= 5196.*ALOG(TK/273.16)-2078.*ALOG(1.E-5*PN)+783.*(1/TK**1.33ESHE0149$
$     133-12580/TK**3)*1.E-5*PN+4.04E-2                                  ESHE0150$
$  999 RETURN                                                            ESHE0151$
$       END                                                              ESHE0152$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
 SSSSSSSSSS    TTTTTTTTTTTT    OOOOOOOOOO    PPPPPPPPPP
SSSSSSSSSSSS   TTTTTTTTTTTT   OOOOOOOOOOOO   PPPPPPPPPPP
SS          SS      TT       CC          OO   PP         PP
SS                  TT       CO          OO   PP         PP
SSS                 TT       OO          OO   PP         PP
 SSSSSSSSS          TT       OO          OO   PPPPPPPPPPP
  SSSSSSSSS         TT       OO          OO   PPPPPPPPPP
        SSS         TT       OO          OO   PP
         SS         TT       OO          OO   PP
SS       SS         TT       OO          OO   PP
SSSSSSSSSSS         TT       OOOOOOOOOOOO   PP
 SSSSSSSSSS         TT        OOOOOOOOOO    PP
```

DATE 04.08.70 TIME 04.07.40

IN MAPLIB ARE INTEGRATED

121 FUNCTIONS

14 MATERIALS

23 PROPERTIES

LATEST CHANGE OF MAPLIB: 24.07.70 AT 01.32.50

FREE REGION : 18 K BYTES

NO ERRORS WERE NOTICED IN THIS JOB BY MAPLIB-UTILITY

A n h a n g 9

MAPLIB-Utility ruft einige Subroutinen auf, die nicht als FORTRAN-Programme vorliegen:

1) Subroutine DATUM

Aufruf: CALL DATUM (DAT,UHR)

Parameter: REAL*8 DAT,UHR
Nach einem Aufruf von DATUM enthalten DAT und UHR das aktuelle Datum bzw.die Uhrzeit in der Form von jeweils 8 Zeichen

Beschreibung: Chr.Hinze, IBM, Programmbeschreibung Nr.199 KFZ Karlsruhe(1969)

2) Subroutine FSPIE

Aufruf: CALL FSPIE

FSPIE liefert Diagnostik-Hilfen im Falle eines Programmausführungsfehlers.

Beschreibung: J.Enzmann, Programmbeschreibung Nr.209 KFZ Karlsruhe (1970)

3) Subroutine FREESP

Aufruf: CALL FREESP(I)

Parameter: I

FREESP ermittelt den vom Programm innerhalb der verfügbaren REGION nicht belegten Kernspeicherplatz. Nach dem Aufruf gibt I die Zahl der freien Speicherplätze in K Bytes (1024 Bytes) an.

Beschreibung: Chr.Hinze, IBM, Programmbeschreibung Nr.210 KFZ Karlsruhe(1970)

4) Subroutine A8FØRM

mit Entries RAMANF und RAMEND

Aufruf: CALL A8FØRM(IA,R8)

Parameter: INTEGER IA, REAL*8 R8
IA ist die Druckeinheit
R8 enthält 8 Zeichen, die entsprechend dem Format A8 mit Großbuchstaben geschrieben werden.

Aufruf: CALL RAMANF(IA)

CALL A8FØRM(IA,R8)

CALL RAMEND (IA)

Bei einem derartigen Aufruf werden die Großbuchstaben mit Sternen eingerahmt.

Beschreibung: Otto Eggenberger, Universität Karlsruhe, Programmbeschreibung Nr.230 KFZ Karlsruhe (Feb.1970)