- Computational Physics -

CT I ***SS 2006***

Blatt 1

Einführung in grundlegende Operationen

1. *Erstellung einfacher Programme*

Erstellen Sie ein *„Begrüßungsprogramm*“. Dieses Programm soll Ihnen beim Aufruf

„guten Morgen“

wünschen.

Anleitung:

In Fortran besteht ein solches Programm aus 2 Zeilen:

```
write (*,*) 'guten Morgen'
end
```

Man mache sich zunächst mit einem der unter Windows verfügbaren Editoren vertraut (z.B. „`proton`“, „`emacs`“ oder „`notepad`“). Damit erzeuge man im Home-Directory (*Platte I: unter Windows (NWZnet)* in einem Unterverzeichnis eine Datei (File) `gruss.f90`. Das darin erstellte Programm kompiliert man dann mit dem Intel-Fortran Compiler (Befehl '`ifort`' im Fortran Command Window „Fortran IA32 Build Environment“) und erzeugt daraus unmittelbar (compile + link) den ausführbaren Code.

Man erstelle ein entsprechendes Programm auch unter C++. Hierfür benutze man den Intel-C++ Compiler unter Windows (Kommando „icl“ im C++ Command Window „C++ IA32 Build Environment“).

Anmerkung: Für die häusliche Nutzung kann ein freier Fortran-Compiler heruntergeladen werden:

http://www.salfordsoftware.co.uk/compilers/support/downloads.html

Sofern man einen guten Internetzugang zur Universität hat, empfiehlt es sich allerdings, über den Terminalserver „nwzho-me.uni-muenster.de“ im NWZnet einzuloggen.

2. Berechnung einfacher Funktionen

a) Man erstelle ein einfaches Programm (in FORTRAN) zur Berechnung von *Fakultäten*

$$n! = \prod_{i=1}^{n} i$$

Hiermit berechne man:

$$n_1 = 34! \qquad \text{und} \qquad n_2 = 35!$$

Man überlege sich, inwieweit es sinnvoll ist, die Rechnung mit *Integer*-Variablen auszuführen. Gibt es eine größte Zahl, für die Sie die Fakultät noch berechnen können, und wie lautet diese?

b) Erstellen Sie ein Programm zur Berechnung von **Summen**

5. Parameter-Übergabe bei Unterprogrammen

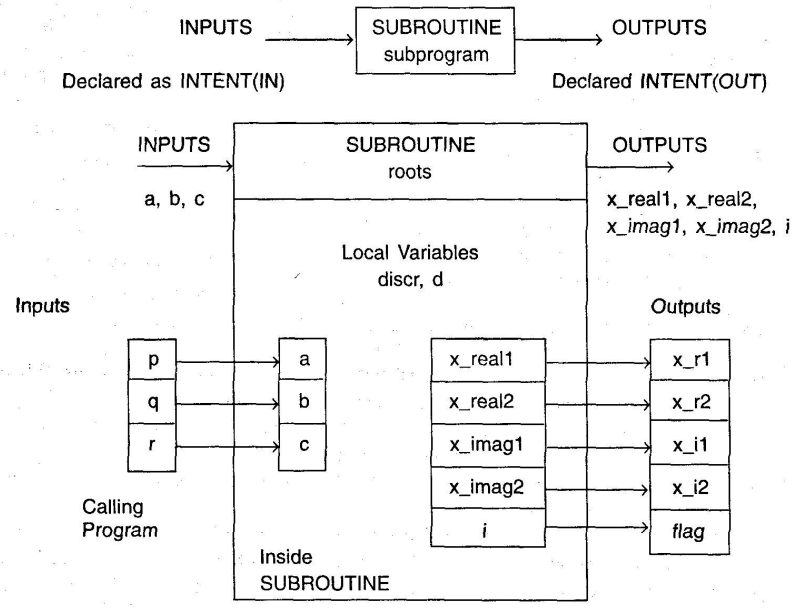

Fig. 9.2 Inputs and outputs of SUBROUTINE subprogram.

$$a_n = \sum_{i=1}^{n} i$$

für verschiedene Werte von n.

Wählen Sie z. B. n = 10^2, 10^3, 10^4.

c) Entwickeln Sie Programme zur Verarbeitung von **Vektoren**.

Gesucht wird das *Skalarprodukt* zweier n-dimensionaler Vektoren A und B.

Anleitung:

Man entwickle ein allgemeines Programm zur Berechnung von Skalarprodukten. Dabei sollen die Dimension n der Vektoren A und B und deren Komponenten als Eingangsparameter frei zu wählen sein.

Welches Ergebnis erhält man für:

$$\mathbf{A} = (0.2, -0.75, 1, 1, -5.3)$$

und

$$\mathbf{B} = (-0.75, 0.02, 1, 5.3, 1)$$

d) Man entwickle entsprechend ein Unterprogramm zur Berechnung des Vektorproduktes:

$$\mathbf{A} = \mathbf{E} \times \mathbf{B}$$

Wobei $\mathbf{E}$ und $\mathbf{B}$ jeweils dreidimensionale Vektoren sind, mit

$$\mathbf{E} = (2,3,1) \qquad \text{und} \qquad \mathbf{B} = (1,2,5)$$

e) Entsprechend entwickle man einfache Programme zur Bearbeitung von ***Matrizen***.
Berechnen Sie das Produkt zweier Matrizen G und H:

$$G = \begin{pmatrix} 1 & 0 & \varepsilon \\ 0 & 1 & 0 \\ -\varepsilon & 0 & 1 \end{pmatrix} \qquad H = \begin{pmatrix} 2 & 0 & z \\ 0 & 1 & 0 \\ z^* & 0 & 1 \end{pmatrix}$$

Es seien $\varepsilon = 1.0\ 10^{-3}$ und $z = 2.5 + 3.0i$.

Auch hier entwickle man zunächst ein allgemeines Programm zur Multiplikation von Matrizen, in dem die Dimension und die Komponenten frei wählbar sind.
Beachten Sie, daß die Matrix H komplex ist!

Zum Vergleich setze man für obige Martixoperationen auch die im Fortran 95 enthaltenen Array-Statements ein (vgl. Anhang).

3. Man erstelle ein ***tageszeitabhängiges Begrüßungsprogramm***.

Man lese hierzu zunächst Datum und Zeit aus dem Computer (der FORTRAN Compiler enthält dafür vorgesehene Systemroutinen) aus. Je nach Tageszeit wähle man den entsprechend formulierten Text .
Das Ergebnis soll formatiert dargestellt werden.

4. Aufruf von Funktions-Unterprogrammen

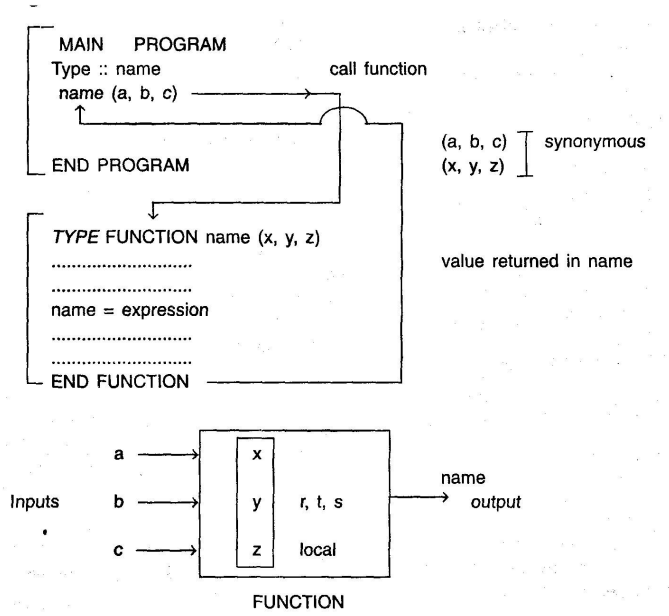

Fig 9.1 Illustrating calling of a function subprogram.

3. Intrinsische Funktionen zur Berechnung von Vektoren und Matrizen

Table 10.5 Intrinsic Functions for Vectors and Matrices

Name	*Arguments*	*Result*
MATMUL (a, b)	Conformable matrices a and b or a matrix and b vector	Matrix product
DOT_PRODUCT (a, b)	a, b vectors	Scalar product = $a_1b_1 + a_2b_2 \ldots a_nb_n$
TRANSPOSE (a)	a matrix	Transpose of a
MAXVAL' (a)	a an array or a(N, i) where N is a constant	Maximum value among all elements of an array
MINVAL (a)	Same as above	Minimum value among all elements of an array
PRODUCT (a)	Same as above	$a_1 * a_2 * a_3 * \ldots\ldots * a_n$
SUM (a)	Same as above	$a_1 + a_2 + a_3 + \ldots\ldots + a_n$

Anhang

1. Struktur eines Fortran Programmes

```
PROGRAM my_first_program
!Purpose:
!To illustrate some of the basic features of a Fortran
! program.

!Declare the variables used in this  program.
INTEGER :: i, j, k !All variables are integers

!Get two values to store in variables i and j
WRITE (*,*) 'Enter the numbers to multiply:'
READ (**) i, j

!Multiply the numbers together
k=i*j

!Write out the result.
!WRITE (*,*) 'Result =', k

!Finish up.
STOP
END PROGRAM my_First_prograrn
```

2. Lesen und Schreiben von Daten in Fortran

(Input/output Format)

Wurde ein File eröffnet (Open-Statement), und ihm dabei die Nummer '`IUN`' zugewiesen, so kann unter dieser Nummer auf den File geschrieben, bzw. von dem File gelesen werden.

```
READ (IUN, ..... )   iolist
WRITE (IUN, ..... )  iolist
```

Optionen:

FMT='format'	Datenformat
END=label	Bei Datenende geht das Programm nach 'label' (nur fuer READ)
ERR=label	Bei Datenfehler geht das Programm nach 'label'
IOSTAT=ios	'ios' ist 0 solange keine Fehler auftritt
REC=irec	Wurde der File für direkten Zugriff (direct access) geöffnet, so wird von/auf Record 'irec' gelesen bzw. geschrieben.

'iolist' ist die Liste der zu schreibenden, bzw. zu lesenden Variablen.

Häufig benutzte Formatspecifier:

`nX`	n Leerstellen
`Il`	Integer, l: Länge = Zahl der Ausgabepositionen
`Fl.d`	Festkommazahl, l: Gesamtlänge d: Zahl der Dezimalstellen
`El.d`	einfach genau Exponentialdarstellung
`Dl.d`	doppelt genau
`Gl.d`	Mischung aus F und D/E
`A`	Characterstring - Längenangabe nicht erforderlich

Zu beachten: Die erste Position in der Druckausgabe dient zur Steuerung des Zeilenvorschubs

`leer`	: es wird eine Zeile weitergeschaltet
`0`	: es werden zwei Zeilen weitergeschaltet
`1`	: es wird eine Seite weitergeschaltet
`+`	: kein Zeilenvorschub

Beispiel: Ausgabe eines komplexen Vektors

```
      IMPLICIT REAL*8 (A-H,O-Y)
      IMPLICIT COMPLEX*16 (Z)
      PARAMETER (N=25)
      DIMENSION Z(N)
      …
      OPEN (6,FILE='PRN')

      …

      DO I=1,N
        WRITE (6,'(1X,A,I2,2(D16.10))')
     &       'I= ',I,' z = ',Z(I)
      END DO
      …
```