FORTRAN90 文法の基礎

2000／1／5　泉　聡志

2000／5／2　泉　聡志

DO 文

- outer: do →　end do outer など、do 構文名をつけることができる。 P43
- exit 文は do 構文の終了 (exit outer)p44
- cycle 文は do ブロックの残り部分の飛び越し (cycle outer)　P44
- do 文の後に何も書かなければ永久に繰り返す。

CASE 文

- select case (場合式)
  case 場合選択子
  end select　　p41

配列

- 2 次元配列の順序は(1,1)(2,1),(1,2),(2,2),(1,3),(2,3)　下向きが一次元　p64
- real,dimension(:)　:: D(m)　で配列宣言　p67
- A=1.0 で A の要素すべてが 1.0 になる。P67
- A(2,:)=-A_0 で:部がすべて A_0 になる。
- C=A+B、C=A*B　で要素ごとの和・積が計算できる。
- sum(A)、dot_product(A,B)で 配列の要素の和、内積が計算できる。
- matmul(A,B)は行列の積　p76-79

割り付け配列

- real,dimension(:,:),allocatable::A　！ 割り付け配列　　p75
  read(*,*)n
  allocate(A(n,n), stat=ier)　　：割付
  if (ier /= 0) exit　　：割付が成功したら ier=0
- allocatable ::　nkind(:)　！ 割り付け配列
  allocate (nkind(im))

文字型

- len( ), index(c,'PQ')は、文字列の長さと'PQ'が現れる位置を示す組み込み関数 p99
- ichar は文字を 10 進数に変換　p106 など

構造型

- type kodomo　　p107
  character(len=8)　namae
  integer　　nenrei
  real　　shincho
  end type kodomo

文字型、整数型、実数型が組み合わさった構造型、
type(kodomo) chonan　　**で変数を宣言、**chonan%shincho **で成分表示する**

ポインタ

- integer,pointer ::p1,p2　**でポインタ宣言**　　p110
- integer,target :: t1,t2 **でターゲット宣言**
- allocate(p1)**でポインタの割り付け。暗黙的に** target **属性を持つ実体**(p1)**を生成する。**
- p1=> t1 **でポインタ代入**
- ASSOCIATED(pointer [,target])**は** pointer **の結合状態を調べる。**
- **ポインタの連鎖**(1)　　p112
  データの総数が不明であり、入力が終わらないと配列の割り付けができない場合に連鎖が威力を発揮する。
  図 5.9(p113)**参照**

```
type cell
  integer       :: value
  type(cell),pointer   :: link
end type cell
type(cell),pointer:: p1,p2
allocate(p1);    p1%value=10
do I=1,10
  read(*,*)k
  allocate(p2)
  p2 = cell(k,p1)     ! 連鎖の形成
  p1=>p2              !  p1 を連鎖に結合
end do
```

- **ポインタの連鎖**(2)
  ポインタ配列は大きさを指定する必要がなく、右辺の配列の大きさを左辺に引き渡せる(p114)。
  必ず、ポインタの代入を行ったあとに、ターゲットに値を入れる。

```
seiseki(i)%kamoku=>ten_su
```

read(*,*)ten_su

- **ポインタの連鎖**(3)

  p142 **の例題** 6.9 **が重要な例題！セルの生成、削除を行っている。**

モジュール（module）

- **宣言、データの初期値、内部副プログラム**(contains **を使用)を含む構成。**Use **文によって参照結合される。宣言は** common **文**(public **属性（参照許可属性）→反対は** private) **と同じ効果がある。**P139
- interface **文と** module procedure **によって、利用者定義の演算子が作成可能** p145
- allocate **配列を使う場合の例**

```
MODULE WORK ARRAYS      ! work array
INTEGER N
REAL,ALLOCATABLE::A(:),B(:,:),C(:,:,:)
END MODULE WORK ARRAYS

PROGRAM GLOBAL_WORK
 PRINT*,' START OF JOB '
 CALL CONFIGUR_ARRAYS
 CALL COMPUTE
END

SUBROUTINE CONFIGUR_ARRAYS
USE WORK ARRAYS
READ(5,*) N
 ALLOCATE(A(N),B(N,N),C(N,N,N))
C
DO 10 I=1,N
 DO 20 J=1,N
   B(I,J)=I+J
20   CONTINUE
10  CONTINUE
END

SUBROUTINE COMPUTE
USE WORK ARRAYS
C
DO 10 I=1,N
 DO 20 J=1,N
  A(I)=A(I)+B(I,J)
20   CONTINUE
10  CONTINUE
PRINT*,' A=',(A(I),I=1,N)
END
```

入出力文

- **一行ずつ文字として読み取って数値に変換が可能（内部ファイル）**p159

  その場合 read(*,'(a)')digit ; read(digit,'(5x,i1)')p1(0) **のように読み込む**

- **ファイル操作** p167

  read(*,'(a)') in_file ; L=index(in_file, '.') ; out_file=in_file(1:L)//'txt'

open(20, file= out_file)

- inquire (file = 'calib.d', exist = ex) ! calib.d **が存在すれば**　ex=.true.　p173

宣言文

- external **文**
  実引数を手続き名にするときに必要。仮変数を手続き名にする際も必要。これを仮手続きと呼ぶ。
- save **文**
  副プログラムにおいて宣言した実体が実行後も保持される。Save **がなくても保持されるが保証はされない。**
- intent **文** p147　　program(**実引数**)→subroutine(**仮引数**)
  intent(in),intent(out)**は実／仮引数の受け渡しの方向を指示する。**(inout)**で双方向**
- intrinsic **文**　p189
  組み込み関数を実引数にする際に使用。仮手続きは external **文を使用する。**
- entry **文**　p190
  同じサブルーチンの中で入口を変えることができる。宣言は前部にまとめて書く。
- **再帰手続き（**recursive**）**　p194
  再帰手続きをしたプログラムを呼び出すと、副プログラムは分身を生成し、save **文によって保存される以外のデータは分身で扱う。これにより、再帰的に手続きを引用してもデータの混同が起こらない。**
- **手続き引用仕様**
  手続き名を実引数・仮引数とする場合は、external **文か手続き引用仕様が必要。配列値を返す手続きなどは** external **文が使えない。**

```
interface  ! F の引用仕様
  function  F(n,x,y)      ! 引用仕様本体
     double  precision  x,y(n),F(n)
  end  function
end interface
```

内部副プログラム

- **通常のサブルーチンを外部副プログラム、**contain **文を挿入して、入れ子の形で含むサブルーチンを内部副プログラムと呼ぶ。親プログラムの変数は内部副プログラムで参照可能であり、これを親子結合と呼ぶ。内部副プログラムは親プログラム以外からは参照できない。**

その他

- 注釈は！をつけてその後に
- write(*,*)('-',I=1.35) で線が引ける。
- data (A(1,j),j=1,n)/1.1,1.2,1.3,1.4,1.5/ でデータ指定
- kind( ) は種別型パラメタを返す問い合わせ関数　kind(1.0)=4, kind(1.0d0)=8
  倍精度の指定は real(kind(1.0d0)　で可能　　p87
  基本論理型の種別型パラメタは 4 になる。
- 形状明示配列・大きさ引継ぎ配列　p124
  形状明示配列は subroutine での配列宣言において大きさを明示する。
  大きさ引継ぎ配列は dimension x(*)とする。明示したほうが無難
- 組み込みサブルーチン：data_and_time(data,time,zpme,values)
- include 文は FORTRAN77 の手法、モジュールの導入によって use 文で代替できる。
-