

フォートラン(fortran)

フォートランは、それほど難しい言語ではありません。変数宣言も強制ではないので慣れればプログラミングが速くなります。一番の特徴は、変数の頭文字でその変数の型が決まることでしょうか。また、左側をラベル用に7文字分空けることも大きな特徴です。

コメント

行の1文字目に「c」を書く、または、任意文字目に「!」を書くとコメントになります。

```
c      コメント1  
      !コメント2
```

変数の型宣言

最初に、変数の型宣言を行います。

```
implicit real precision(a-h,o-z)
```

上記は、変数の頭文字が、a-h, o-z の場合、データ型が real 型になり、それ以外の変数は integer 型になるという意味です。

```
hoge :real 型
```

```
ij,k :integer 型
```

となります。倍精度の変数を使用する場合は、下記を記述します。

```
implicit double precision(a-h,o-z)
```

配列宣言

配列は、次の様に宣言します。

```
dimension ary1(499)
```

```
dimension ary2(499,499)
```

parameter 文

parameter 文は、定数を宣言する場合に使用します。

```
implicit double precision(a-h,o-z)
```

```
parameter(ind=499)
```

```
dimension ary(ind)
```

data 文

data 文も、定数を宣言する場合に使用します。

```
implicit double precision(a-h,o-z)
```

```
data para1,para2/1,2/
```

function 関数

関数を宣言する命令文です。

```
implicit double precision(a-h,o-z)
i=1
j=2
k=add(i,j)
write(*,*)k
stop
end
```

```
function add(i,j)
implicit double precision(a-h,o-z)
add=i+j
return
end
```

出力結果

3

subroutine 関数

関数を宣言する命令文です。

```
implicit double precision(a-h,o-z)
i=1
j=2
call add(i,j,k)
write(*,*)k
stop
end
```

```
subroutine add(i,j,k)
implicit double precision(a-h,o-z)
k=i+j
return
end
```

出力結果

3

common 文

common 文は、関数間で変数を共有する場合に使用します。

```
implicit double precision(a-h,o-z)
input()
output()
stop
end
```

!入力用関数

```
subroutine input()
  implicit double precision(a-h,o-z)
  common /data1/i,j
  i=1
  j=2
  return
end
```

!出力用関数

```
subroutine output()
  implicit double precision(a-h,o-z)
  common /data1/i,j
  write(*,*)i,j
  return
end
```

出力結果

1 2

include 文

ファイルを読み込む命令文です。使用する配列等をまとめて宣言しておくとう便利です。

```
include "head.for"
call hoge()
stop
end
```

```
subroutine hoge()
include "head.for"
```

head.for

```
implicit double precision(a-h,o-z)
dimension ary1(499)
dimension ary2(499,499)
```

do 文

do 文は繰り返しを行う命令文です。

```
i=0
do i=1,49
  i=i+1
enddo
```

if 文

if 文は条件分岐の命令文です。

```
i=1
j=2
if(i.eq.1 .and. j.eq.2)then
  write(*,*)i
elseif(i.ne.1 .or. j.ne.2)then
  write(*,*)j
else
  write(*,*)i,j
endif
```

条件の種類は下記となります。

eq:等しい

le:以下

lt:未満

ge:以上

gt:より大きい

and:かつ

or:または

cycle、exit 文

cycle 文は、ループの先頭に移動する命令文です。exit 文はループを終了する命令文です。

```
do i=1,10
  if(i.lt.5) cycle
  exit
enddo
write(*,*)i
```

出力結果

5

ファイル操作

ファイルから読み込む

```
open(10,file="hoge.inf")
do i=1,10
  read(10,*)ary(i)
enddo
close(10)
```

ファイルに書き込む

```
open(10,file="hoge.inf")
do i=1,10
  write(10,*)ary(i)
enddo
close(10)
```

goto 文

goto 文は下記の様に使用します。

```
do 100 i=1,10
  if(i.eq.5)goto 200
100  continue
200  continue
```