情報処理概論(第5回)

情報基盤センター　井上 仁

授業内容(第5回)

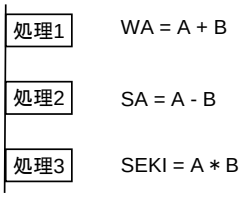
- Fortranの文法の説明
  - ◆IF構文(続き)
    - ■IF THEN文、ELSE文 、ELSE IF文 、END IF文
  - ◆CASE構文
    - ■SELECT CASE文、CASE文、END SELECT文
  - ◆DO構文
    - ■DO文、END DO文、EXIT文

プログラムの基本制御構造

- 連接
  - ◆順に処理する
- 選択
  - ◆条件によって処理を選択する
- 反復
  - ◆一定回数、あるいは条件を満たすあいだ処理を繰り返す

連接

- 順に処理する



IF構文(1)

IF (論理式) THEN
　ブロック　(論理式が真のとき実行)
ENDIF

(例)

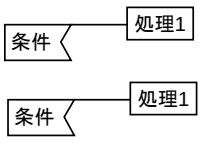
```
IF (b.NE.0) THEN
  sho=a/b
ENDIF
```

ブロックは、複数の実行文

```
IF (b.NE.0) THEN
  WRITE(*,*) 'b is not 0'
  sho = a / b
ENDIF
```

選択(1)

- 条件によって処理を選択する




```
IF (B.NE.0) SHO = A / B
```



```
IF (B.NE.0) THEN
  SHO = A / B
ENDIF
```

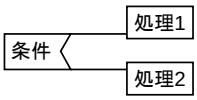
条件によって、処理1が実行されるか、実行されない

IF構文(2)

```
IF (論理式) THEN
   ブロック1  (論理式が真のとき実行)
ELSE
   ブロック2  (論理式が偽のとき実行)
ENDIF
(例)
    IF (b.NE.0) THEN
        WRITE(*,*) 'wa=',wa,' sa=',sa,' seki=',seki,' sho=',sho
    ELSE
        WRITE(*,*) 'wa=',wa,' sa=',sa,' seki=',seki
    ENDIF
```

選択(2)

■条件によって処理を選択する




```
IF (B.NE.0) THEN
   WRITE(*,*) ……
ELSE
   WRITE(*,*) ……
ENDIF
```

条件によって、処理1が実行されるか、処理2が実行される

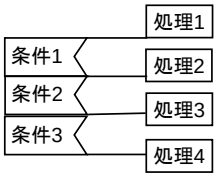
IF構文(3)

```
IF (論理式1) THEN
   ブロック1  (論理式1が真のとき)
ELSE IF (論理式2) THEN (論理式1が偽のとき)
   ブロック2  (論理式2が真のとき)
……
ELSE IF (論理式n) THEN (論理式n-1が偽のとき)
   ブロックn  (論理式nが真のとき)
ELSE
   ブロックn+1  (論理式nが偽のとき)
ENDIF
```

IF構文の例

```
IF (point >= 80) THEN
    WRITE(*,*) 'A'
ELSEIF (point >= 70) THEN
    WRITE(*,*) 'B'
ELSEIF (point >= 60) THEN
    WRITE(*,*) 'C'
ELSE
    WRITE(*,*) 'D'
ENDIF
```

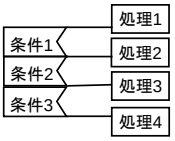
選択(3)




```
IF (point >= 80) THEN
   WRITE(*,*) 'A'
ELSEIF (point >= 70) THEN
   WRITE(*,*) 'B'
ELSEIF (point >= 60) THEN
   WRITE(*,*) 'C'
ELSE
   WRITE(*,*) 'D'
ENDIF
```

条件によって、処理1が実行されるか、処理2が実行されるか、処理3が実行されるか、処理4が実行される。

選択(3')




```
IF (point >= 80) THEN
   WRITE(*,*) 'A'
ELSEIF (point >= 70) THEN
   WRITE(*,*) 'B'
ELSEIF (point >= 60) THEN
   WRITE(*,*) 'C'
ELSE
   WRITE(*,*) 'D'
ENDIF
```

```
SELECT CASE (point)
CASE(80:)
    WRITE(*,*)  'A'
CASE(70:79)
    WRITE(*,*)  'B'
CASE(60:69)
    WRITE(*,*)  'C'
CASE DEFAULT
    WRITE(*,*)  'D'
END SELECT
```

CASE構文(1)

SELCECT CASE (式)
CASE 選択子1
　ブロック1
CASE 選択子2
　ブロック2
　.....
END SELECT

CASE構文(2)

■選択子は、値や範囲の並びを括弧でくくったもの
◆並びは、コンマ(,)で区切ったもの
◆範囲は、下限と上限をコロン(:)で区切ったもの
◆DEFAULTは、どの選択子にも含まれないすべての値

(例)

CASE (1)	1
CASE (2:5)	2, 3, 4, 5
CASE (7:)	7, 8, 9, 10, 11, ･･･
CASE (:9)	･･･, 6, 7, 8, 9
CASE (1, 2:5, 10)	1, 2, 3, 4, 5, 10
CASE DEFAULT	

CASE構文(3)

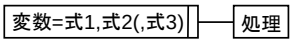
```
SELECT CASE (point)
CASE(80:)
     WRITE(*,*)  'A'
CASE(70:79)
     WRITE(*,*)  'B'
CASE(60:69)
     WRITE(*,*)  'C'
CASE DEFAULT
     WRITE(*,*)  'D'
END SELECT
```

演習5-1

■点数を入力して、成績を出力するプログラムを作成しなさい(IF構文、あるいはCASE構文を用いる)。

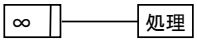
反復

■一定回数、あるいは条件を満たすあいだ処理を繰り返す



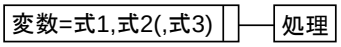
式1:始値, 式2:限界値, 式3:増分値

```
DO i = 1, 10
   処理
END DO
```




```
DO
   処理
END DO
```

DO構文(1)




```
DO 変数 = 式1, 式2, 式3
   処理
END DO
```

式1: 始値
式2: 限界値
式3: 増分値

(1) 変数 = 始値
(2) 変数と限界値の比較
　増分が+で、変数＞限界値
or 増分が-で、変数＜限界値
　⇒ END DOの次の文へ(終了)
　それ以外
　⇒ END DOまで実行
(3) 変数に増分値を加えて(2)へ

DO構文の例(do.f90)

■ iの値が1から3の間繰り返す

```
PROGRAM do

INTEGER :: i

DO i = 1, 3
   WRITE(*,*) i, '回目'
END DO
STOP

END PROGRAM do
```

実行例

```
1 回目
2 回目
3 回目
```

DO構文の例(do1.f90)

■ iの値が1からnの間繰り返す

```
PROGRAM do1

INTEGER :: i, n

READ(*,*) n
DO i = 1, n
   WRITE(*,*) i, i**2
END DO
STOP

END PROGRAM do1
```

実行例

```
4

1   1
2   4
3   9
4   16
```

DO構文の例(do2.f90)

■ iの値が1からnの間繰り返す(iの増分は2)

```
PROGRAM do2

INTEGER :: i, n
READ(*,*) n

DO i = 1, n, 2
   WRITE(*,*) i, i**2
END DO
STOP

END PROGRAM do2
```

実行例

```
4

1   1
3   9
```

演習5-2

■ do.f90, do1.f90 と do2.f90 を入力して、実行しなさい。

DO構文の入れ子(1)

■ 九九の表を作る…

■ まずは、九九の一覧を作る…

	1	2	…	9
1	1	2		9
2	2	4		18
…				
9	9	18		81

```
1 × 1 = 1
1 × 2 = 2
……
1 × 9 = 9
2 × 1 = 2
2 × 2 = 4
……
9 × 1 = 9
9 × 2 = 18
……
9 × 9 = 81
```

DO構文の入れ子(2)

```
DO 変数1 = 式11, 式12, 式13
   DO 変数2   = 式21, 式22, 式23
      処理
   END DO
END DO

DO i = 1, 9
   DO j = 1, 9
      WRITE(*,*) i, '*', j, '=', i*j
   END DO
ENDO DO
```

```
i    j   i*j
1 × 1 = 1
1 × 2 = 2
……
1 × 9 = 9
2 × 1 = 2
2 × 2 = 4
……
9 × 1 = 9
9 × 2 = 18
……
9 × 9 = 81
```

演習5-3

- 九九の一覧を出力するプログラムを作成しなさい。

DO構文(2)


```
DO
   処理
END DO
```

DO構文の例(do3.f90)

- 無限ループ
- EXIT文でDOループを抜ける

```
DO
   WRITE(*,*) 'Input a'
   READ(*,*) a
   IF (a <= 0) EXIT
   WRITE(*,*) a, 'が入力されました'
END DO
```

演習5-4

- do3.f90 を入力して、実行しなさい。

次回の予定(第6回)

- Fortranプログラミング
  - ◆DO構文(続き)
  - ◆配列
  - ◆DO形並び