これまでのまとめ(学年末試験へ向けて)

山本昌志*

2004 年 2 月 24 日

1　ここまでの学習内容

後期の中間試験の後の学習内容は、以下の通りです。教科書で言うと 5 章〜8 章です。これらの使い方をよく理解してください。

- 5 章　組み込み関数
- 6 章　文関数
- 7 章　関数副プログラム
- 8 章　サブルーチン副プログラム

2　組込関数

2.1　どのような時に使うか

FORTRAN は、Formula Translation(数式翻訳) が語源となっているだけあった、技術計算向きのプログラム言語です。技術計算を行う場合、関数の値を求めることは随所に現れます。関数毎に、後で述べる関数副プログラムやサブルーチン副プログラムを作っていたのでは大変です。そこで、FORTRAN には、表 1 のような関数が組み込まれています。これらの数学関数を使うときに、組込関数を使うことで、プログラムが楽になります。

大体のものはおなじみと思いますが、指数関数と自然対数は 2 年生で学習します。剰余は、MOD(A,B) とした場合の A/B の計算の余りのことです。

*国立秋田工業高等専門学校　電気工学科

表 1: よく用いられる組込関数。これ以上のものは、教科書の P.338〜にある。

組込関数	関数名	引数の数	数学の表現
平方根	SQRT	1	$\sqrt{x}$
正弦	SIN	1	$\sin(x)$
余弦	COS	1	$\cos(x)$
正接	TAN	1	$\tan(x)$
逆正線	ATAN	1	$\tan^{-1}(x)$
指数	EXP	1	e^x
自然対数	LOG	1	$\log(x)$
常用対数	LOG10	1	$\log_{10}(x)$
切捨て整数化	INT	1	
絶対値	ABS	1	$\|x\|$
剰余	MOD	2	

2.2 実際のプログラム

組込関数が使われる例として、教科書のプログラム[1]を載せます。このプログラムは、次のような動作をします。

- $\sin^2(x) + \cos^2(x) = 1$ を確認するプログラムです。
- 計算する角度は、0 度から 90 度まで、5 度間隔です。
- 度数、ラジアン、$\sin(x)$、$\cos(x)$、$\sin^2(x) + \cos^2(x)$ の値を並べて、書き出す。

ここで使われている SIN(X) と COS(X) が組込関数です。

文番号 601 の次の行の 6 桁目にアスタリスク (*) があります。6 桁目に何か書くとそれは、その前の行とつながっていると示しています。1 行が長い場合に使います。

```
      REAL R,U,V,Z,X

      WRITE(6,600)
600   FORMAT('    x    R',6X,'U=SIN(R)',7X,'V=COS(R)',7X,
     *       'U*U+V*V')

      X=0.0

    1 R=X*3.141592/180.0
      U=SIN(R)
      V=COS(R)
      Z=U*U+V*V
      WRITE(6,601)X,R,U,V,Z
601   FORMAT(2F5.1,3E15.8)
```

[1] P.104 [例題] SIN(X),COS(X) の計算

```
      X=X+5.0
      IF(X.LE.90.0)GO TO 1

      END
```

組込関数は簡単なのですぐに理解できると思います。ここでひとつ言葉を覚えてもらいます。たとえば組み込み関数 SIN(R) ですが、この変数 R のことをコンピューターの世界では、実引数[2]と言います。関数を呼び出すときの変数のことです。

2.3 組込関数の文法

2.3.1 組み込み関数の呼び出し

関数の呼び出しは、

関数名（実引数並び）

です。要するに普通の数学の関数とほとんど同じです。このようにして呼び出すと、その関数の値が戻ってきます。この戻ってきた値を戻り値と言います。通常は、先の例のプログラムのように、戻り値は他の変数に格納します。

2.3.2 組込関数の定義

プログラマーが定義する必要はありません。組込関数と言われるものは、ちゃんと FORTRAN の中で定義されています。気にしないで、使えます。

2.4 組込関数の注意事項

細かい注意事項はいろいろありますが、皆さんがこれらの組込関数を使う上で気をつけるべきことは、ただひとつ

- 三角関数の引数 (角度) の単位は、弧度法の [rad] です。

と言うことです。

[2]あるいは単に、引数と呼ぶこともある。

3　文関数

3.1　どのような時に使うか

文関数とは、1 行で書く関数です。たとえば、

$$\begin{aligned} f(x) &= x^3 + 5x^2 + 2x + 6 \\ g(x) &= \sin(x) * \cos(x) - x^2 + \tan(5x^3) \end{aligned} \tag{1}$$

のような関数の計算が必要な場合、文関数を使います。x に値を入れたら、計算されるような、通常の関数の場合です。これは 1 変数の関数ですが、2 変数以上の多変数の関数にも使えます。たとえば、

$$\begin{aligned} f(x,y) &= 5x^2 + 3xy + 2y^2 + 6x + 9y + 7 \\ g(x,y,z) &= \sin(x+y) + \cos(y+z) + \tan(x+y+x) + xyz \end{aligned} \tag{2}$$

のような場合です。多変数の関数は、まだ学習していないかもしれませんが、これも 1 変数と同じで、x や y に値を入れると、関数の値がただひとつ決まります。

このように、1 行で定義できるような関数を計算するときに文関数を使います。文関数を用いると、これを呼び出すだけで、何回も計算ができ便利です。

3.2　実際のプログラム

文関数が使われる例として、教科書のプログラム[3]を載せます。このプログラムは、次のような動作をします。

- 2 次方程式 $ax^2 + bx + c = 0$ の 1 根 $(-b + \sqrt{b^2 - 4ac})/(2a)$ を求める分関を定義する。
- 3 組の係数のデータを読み込む。
- 答 1 は、読み込んだ係数 x, y, z について 1 根を求め、印刷する。
- 答 2 は、読み込んだ係数 $2x, 3y, z$ について 1 根を求め、印刷する。

ここで使われている ROOT(A,B,C)=(-B+SQRT(B*B-4.0*A*C))/(2.0*A) が文関数です。

```
      ROOT(A,B,C)=(-B+SQRT(B*B-4.0*A*C))/(2.0*A)

      DO 10 I=1,3
        READ(5,*)X,Y,Z
        ANS1=ROOT(X,Y,Z)
        ANS2=ROOT(2.0*X,3.0*Y,Z)
        WRITE(6,*) 'ANS1=',ANS1,'   ANS2=',ANS2
   10 CONTINUE

      END
```

文関数の使い方は組込関数とほとんど同じです。ただ、関数の定義はプログラマーの仕事です。

[3] P.112 [例題] 2 次方程式の一つの根を求める文関数

3.3 文関数の文法

3.3.1 文関数の呼び出し

文関数の呼び出しは、

関数名 (実引数並び)

です。組込関数と同じで、普通の数学の関数と同じ感覚で使えます。このようにして呼び出すと、その関数の値が戻ってきます。この戻ってきた値を戻り値と言います。通常は、先の例のプログラムのように、戻り値は他の変数に格納します。

3.3.2 文関数の定義

文関数の定義は、宣言文の中で書く必要があります。要するに実行文に先立って、以下のように書きます。まず、関数と引数の型の宣言を行います。そうして、関数の定義を書きます。

```
型 関数名
型 引数名
関数名 (実引数並び)=関数の計算式
```

関数や引数の型宣言を書かなかった場合、暗黙の型宣言[4]に従います。関数と引数の型が同じであれば、同一の行に書いても良いです。ですから、先の例のプログラムは、

```
REAL ROOT
REAL A,B,C
ROOT(A,B,C)=(-B+SQRT(B*B-4.0*A*C))/(2.0*A)
```

と書くべきところを、暗黙の型宣言を利用して、関数の型の宣言を省略しています。

3.4 文関数の注意事項

定義した文関数が使えるのは、定義を行ったメインルーチンやサブルーチンの中だけです。メインルーチンで定義した文関数はメインルーチンでしか使えません。副プログラム (サブルーチン副プログラム、関数副プログラム) で定義した文関数は、それぞれの中でしか使えません。変数と同じです。

4 関数副プログラム

4.1 どのような時につかうか

教科書の例にならい、次のような関数を計算する場合、どうするか考えよう。

$$f(x) = \begin{cases} \sqrt{1-x^2} & -1 \leq x \leq 1 \text{ のとき} \\ 0 & \text{それ以外のとき} \end{cases} \tag{3}$$

[4]関数名が I,J,K,L,M,N で始まる場合、その戻り値は整数になる。

この関数を 1 回のみ計算する場合は、IF 文を使うことで対処できますが、何回も計算する場合は厄介です。通常のプログラムでは似たような計算を大量に行うので、できるだけ同じ処理はまとめたほうが効率的です。先の関数を何億回も計算することは、決して珍しくありません。

そのために、この関数を定義して、いつでも呼び出せるようにすると便利です。あたかも、$\sin(x)$ のように x を入れれば、計算してくれるようにすれば非常に便利です。一つの解決方法が文関数ですが、これは 1 行で関数を定義する必要があるので、式 (3) のように IF 文を使って、場合分けが必要の時には使えません。文関数で定義できない複雑な関数を定義するときに、関数副プログラムを使います。

もう一度、文関数と関数副プログラムの違いをまとめておくと次のようになります。

文関数 1 行で定義できる簡単な関数のときに使います。また、ていぎしたメインルーチン、あるいは副プログラム以外から呼び出すことはできません。

関数副プログラム 複数の行で定義する必要がある複雑な関数のときに使います。

4.2 実際のプログラム

教科書の例[5]を用いて説明しよう。教科書のプログラムは分かりにくいので、実際、通常のプログラマーが記述するように書くと以下のようになります。

メインルーチンを見るだけで、このプログラムの動作が分かると思います。関数 F(X) の値を X=0.5 から 0.1 ステップで 1.5 まで計算して、書き出していると想像できます。実際の関数は、関数副プログラムを見ればすぐに分かるようになっています。

関数 F(X) はどこからでも使うこともでき便利です。同じ処理を何回も書かないですむということです。これは、関数を変えたい場合、プログラムを書き直すのは 1 箇所で済みます。これにより、プログラムは分かりやすくなり、作成時間も非常に短くできます。

```
*=========================================================
* MAIN ROUTINE
*=========================================================
      PROGRAM MAIN

      DO 10 X=0.5, 1.5, 0.1
        Y=F(X)
       WRITE(6,601) X,Y
  601   FORMAT(2E20.8)
   10 CONTINUE

      STOP
      END

*=========================================================
*     USEER DEFINEDED FUNCTION
*=========================================================
      FUNCTION F(X)
```

[5] P.122 [例題] $f(x) = \sqrt{1-x^2}$ の計算

```
      IF(ABS(X).LE.1.0)THEN
        F=SQRT(1.0-X*X)
      ELSE
        F=0.0
      ENDIF

      RETURN
      END
```

4.3 関数副プログラムの文法

4.3.1 関数の呼び出し

関数は、関数名で呼び出して、算術代入文で変数に入れるのが普通です。関数名は、組み込み関数と異なれば、任意の名前をつけることができます。実引数並びは、関数の独立変数で、複数個が許されます。多変数関数に対応しているということです。また、配列も許されます。ただし、実引数と仮引数の数と型は一致させる必要があります。

同様に、関数の戻り値の型と代入される変数の型も一致させる必要があります。

型　関数名
代入される変数=関数名（実引数並び）

関数の計算結果の型は、関数名の型宣言により決まります。もし、型宣言を行わなければ、暗黙の型宣言になります。型宣言を行った関数副プログラムは、次のようになります。

```
*=========================================================
* MAIN ROUTINE
*=========================================================
      INTEGER WA

      I=2
      J=3
      K=WA(I,J)

      STOP
      END

*=========================================================
* FUNCTION
*=========================================================
      INTEGER FUNCTION WA(M,N)

      WA=M+N

      RETURN
      END
```

4.3.2 関数の定義

関数の定義は以下のように行います。関数の型は、REAL や INTEGER 等、変数の場合と同じように書きます。これは、戻り値の型を示します。ただし、これは省略可能で、その場合は関数名に従い暗黙の型宣言[6]になります。

仮引数並びと実引数並びは、同じ数で型でなくてはなりません。ただし、引数の変数名は異なっても、何ら問題はありません。関数の処理の部分は、今まで学習してきた FORTRAN のプログラムを書けばよいのです。計算のみならず、WRITE 文や READ 文も書けます。ただし、最後に関数名に値を代入する必要があります。これが戻り値になります。

関数副プログラム内で使われた変数は、引数以外は全く、メインルーチンに影響を及ぼしません。メインルーチンと同じ名前の変数が使われていても、その結果が反映されることはありません。

```
関数の型 FUNCTION 関数名 (仮引数並び)
宣言文 (仮引数などの宣言)

実行文 (関数の値を求めるための処理)

関数名=処理の結果の関数の値

RETURN
END
```

4.4 関数副プログラムの注意事項

呼び出し側の変数と関数副プログラムの変数は、引数を除いて完全に独立です。引数は、その並び方の順序で、呼び出した側と呼ばれた側で完全に一致します。これを、以下の例をもって、示します。

```
*=========================================================
* MAIN ROUTINE
*=========================================================
      INTEGER WA

      I=2
      J=3
      K=4
      M=WA(I,J)

      WRITE(6,601)I,J,K,M
  601 FORMAT('I=',I2,'  J=',I2,'  K=',I2,'  M=',I2)

      STOP
      END

*=========================================================
* FUNCTION
*=========================================================
```

[6]関数名の始まりが、I, J, K, L, M, N の時は INTEGER とみなされる

```
      INTEGER FUNCTION WA(M,N)
      INTEGER I,J,K,M,N

      I=99
      J=99
      K=M+N
      WA=K
      M=88
      N=77

      RETURN
      END
```

このプログラムを実行するとその結果は、

```
I=88  J=77  K= 4  M= 5
```

となります。要するに、引数を除いて、メインルーチンと関数副プログラムの変数の格納領域は異なるということです。名前が異なっていても、実引数と仮引数は同じ領域にデータが格納されます。一方、その他の変数は名前が同じでも、異なった領域にデータは格納されます。

5　サブルーチン副プログラム

5.1　どのような時につかうか

関数副プログラムの場合、入力の変数は複数個でも良いが、出力 (戻り値) は 1 個に限ります。それは文法から明らかでしょう。それに対して、サブルーチン副プログラムは、複数の値を返すことができます。

5.2　実際のプログラム

教科書の例[7]を用いて説明しよう。教科書のプログラムは分かりにくいので、実際、通常のプログラマーが記述するように書くと以下のようになります。このプログラムでは、パイプの外径と内径、長さ、密度を与えて、体積と重さを計算しています。それを、SUBROUTINE PIPE で計算しています。

最初の CALL 文では、キーボードから入力された、外径 (A) と内径 (B)、長さ (L1)、密度 (G1) を使って、体積 (V) と質量 (W) を計算しています。

2 番目の CALL 文では、直接サブルーチンに値を与えています。すなわち、外径を 8.0、内径を 6.0、長さを 350.0、密度を 7.8 として、体積 (V) と質量 (W) を計算しています。

```
*=========================================================
* MAIN ROUTINE
*=========================================================
      PROGRAM MAIN
```

[7] P.137〜138 [例題] 管の体積と重さを計算するサブルーチン I

```
      REAL A, B, L1, G1, V, W

      READ(5,*)A, B, L1, G1
      CALL PIPE(A, B, L1, G1, V, W)
      WRITE(6,600)V,W

      CALL PIPE(8.0, 6.0, 350.0, 7.8, V, W)
      WRITE(6, 600)V,W

  600 FORMAT('V=',E15.8, ' W=', E15.8)

      STOP
      END

*=========================================================
*     SUBROUTINE OF PIPE CALCULATION
*=========================================================
      SUBROUTINE PIPE(X, Y, L, G, V, W)
      REAL X, Y, L, G, V, W

      V=3.14159/4.0*(X*X-Y*Y)*L
      W=V*G

      RETURN
      END
```

5.3 サブルーチン副プログラムの文法

5.3.1 サブルーチンの呼び出し

サブルーチンは、CALL サブルーチン名で呼び出します。データの受け渡しは、引数で行います[8]。関数副プログラム同様、実引数並びは、複数個が許されます。また、配列も許されます。ただし、実引数と仮引数の数と型は一致させる必要があります。

CALL サブルーチン名 (実引数並び)

5.3.2 サブルーチンの定義

サブルーチンでの処理の定義は以下のように行います。すなわち、SUBROUTINE と最初に書いて、続いて、サブルーチン名、仮引数が並びます。それから、通常のプログラムのように書いて、RETURN と END 文で終わります。

仮引数並びと実引数並びは、同じ数で型でなくてはなりません。ただし、引数の変数名は異なっても、何ら問題はありません。関数の処理の部分は、今まで学習してきた FORTRAN のプログラムを書けばよいの

[8]引数以外にデータの受け渡しは可能ですが、ここでは述べません。DATA 文等の方法があります。

です。計算のみならず、WRITE 文や READ 文も書けます。

サブルーチン副プログラム内で使われた変数は、引数以外は全く、メインルーチンに影響を及ぼしません。メインルーチンと同じ名前の変数が使われていても、その結果が反映されることはありません。

```
      SUBROUTINE サブルーチン名 (仮引数並び)
      宣言文 (引数や変数などの宣言を行う)

      実行文 (関数の値を求めるための処理)

      RETURN
      END
```

5.4 サブルーチン副プログラムの注意事項

呼び出し側の変数とサブルーチン副プログラムの変数は、引数を除いて完全に独立です。引数は、その並び方の順序で、呼び出した側と呼ばれた側で完全に一致します。これを、以下の例をもって、示します。

```
*=========================================================
* MAIN ROUTINE
*=========================================================
      INTEGER I,J,K,M

      I=2
      J=3
      K=4
      CALL WA(I,J,M)

      WRITE(6,601)I,J,K,M
  601 FORMAT('I=',I2,'  J=',I2,'  K=',I2,'  M=',I2)

      STOP
      END

*=========================================================
* SUBROUTINE
*=========================================================
      SUBROUTINE WA(M,N,K)
      INTEGER I,J,K,M,N

      I=99
      J=99
      K=M+N
      M=88
      N=77

      RETURN
      END
```

このプログラムを実行するとその結果は、

```
I=88  J=77  K= 4  M= 5
```

となります。要するに、引数を除いて、メインルーチンとサブルーチンの変数の格納領域は異なるということです。名前が異なっていても、実引数と仮引数は同じ領域にデータが格納されます。一方、その他の変数は名前が同じでも、異なった領域にデータは格納されます。

6　効率よく勉強する方法

6.1　組込関数、文関数、副プログラムが使われる理由

組み込み関数が使われる理由は明らかです。それは、

- 数学で使われる関数の計算をしたい場合、それを使えば簡単

だらかです。通常の関数のように使えるので便利です。
文関数が使われるの理由は、

- 簡単な、1 行で書ける関数であれば、組み込み関数のように文関数定義し、使える。何回もその関数を使う場合に便利です。
- 関数の形を書き換えるときに簡単です。
- プログラムが分かり易くなる。

などです。
関数副プログラムが使われる理由は、

- 文関数よりも複雑な関数が扱える。複数行で関数が定義できる。
- 同じ関数を複数回使うと時に、1 回の定義ですむ。このようにすると、関数の定義の変更も容易です。
- プログラムが分かり易くなる。

などです。
関数副プログラムが使われる理由は、

- 複雑な処理を機能のブロック単位に分割できる。このことによりプログラムが分かりやすくなる。
- 同じ処理を複数回使うと時に、1 回の処理内容の記述ですむ。このようにすると、処理内容の変更も容易です。
- プログラムが分かり易くなる。

などです。

6.2 プログラム

教科書の例題

- P.103〜104 SIN(X),COS(X) の計算
- P.111〜112 2 次方程式の一つの根を求める文関数
- P.112 $f(x) = \sqrt{1-x^2}$ の関数の計算
- P.137〜138 管の体積と重さを計算するサブルーチン I

を理解すること。特に、ここで学習した組み込み関数と文関数、関数副プログラム、サブルーチン副プログラムの使い方を理解すること。

この教科書の例題とあわせて、授業で学習した練習問題も理解すること。

- 組込関数の練習問題　2 倍角の公式
- 文関数の練習問題　円の面積 (数値積分の台形公式)
- 関数副プログラムの練習問題
- サブルーチン副プログラムの練習問題 三角関数の計算

6.3 練習問題

以下、教科書の練習問題もきちんと理解すること。

- 5 章 (組込関数)
  - 5-A(1),(2),(3),(5)
- 6 章 (文関数)
  - 6-A(1),(2)
  - 6-B(1) これは先に示した授業で学習したプログラムと同じ
- 7 章 (関数副プログラム)
  - 7-A(1),(2)
- 8 章 (サブルーチン副プログラム)
  - 8-A(1)