Fortran

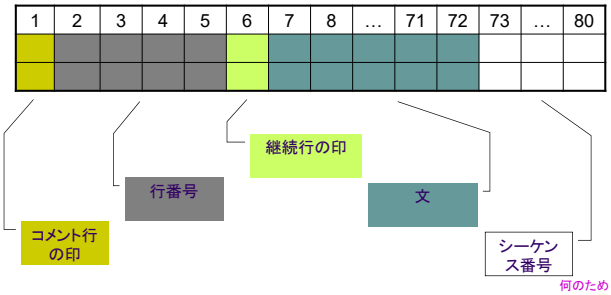
プログラミング言語論

前置きを少し

- IBM, 50年代, “FORmula TRANslator”
- 数値計算、科学技術計算
- コンパイル言語。インタプリタ言語ではない
  - 当初、一行ごとにコンパイルできるよう、工夫された
- Fortran77, Fortran90, Fortran95…
  - 標準化が積極的に進められた プログラマのため? コンパイラのため?

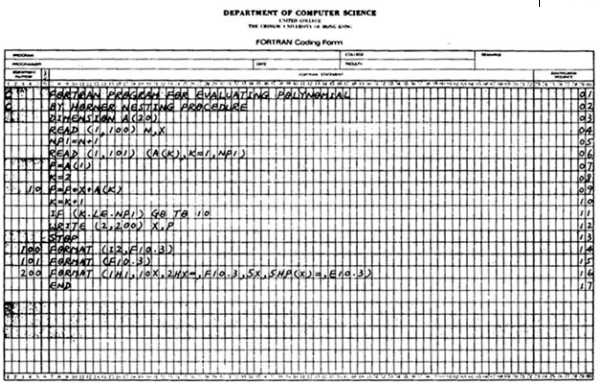
文の書式

- 書式は 固定!!!!!! 文字を書く絶対位置に意味がある



何のため？

こんなものを書いていました



一般的注意を一言

- 大文字小文字の区分なし case insensitive
  - 普通は大文字. 昔は大文字しかなかった
- コメント
  - 第一コラムに文字を書く。通常は ‘C’ または ‘*’
- 空白は無視される（理由不明、是非に議論あり）
- 一行に一文!!
  - セミコロンなし
  - 継続行の印 (何でもよい, 普通 ‘+’ かな)

NASA の事故（という都市伝説）

- ロケット打上げ失敗

```
DO 15 I = 1,100      ←  書いたつもり
DO 15 I = 1.100      ←  タイプした
DO15I=1.100          ←  コンパイラにとって
DO 15,I = 1,100      ←  （言語仕様上）安全策の例
```

Wikipediaのマリナー1号の項を参照してください

次も参照のこと
http://catless.ncl.ac.uk/Risks/9.54.html#subj1
http://catless.ncl.ac.uk/Risks/8.75.html

プログラムの構造

- 構文

```
PROGRAM 名称
      変数宣言
      文
STOP
END
```

変数宣言

- 変数名
  - 最大 6 文字
  - 英数字 (開始は英字. 大文字のみであった)
- 型
  - INTEGER, REAL, DOUBLE PRECISION, COMPLEX, LOGICAL, CHARACTER
  - 例

```
REAL X, Y
```

変数宣言

- 暗黙の型宣言 (先頭文字種による) 宣言がないとき
  - A から H, O から Z: REAL
  - I から N: INTEGER
- 文字列宣言
  - CHARACTER*20 STR
- 定数
  - PARAMETER (PI = 3.14, SIZE = 2)

式と代入

- 論理定数: .TRUE. と .FALSE.
- 文字列: 'I am a string'
- 部分文字列: STR(*start*:*end*)
- 算術式
  - +, -, *, /, **, ()
  - 型変換: 型の混合演算はなし
  - INT(), REAL(), DBLE()
    - X = DBLE(Y) * DBLE(Z)

式と代入

- 関係演算子
  - .EQ., .GE., .GT., .LE., .LT., .NE.
- 論理演算子
  - .NOT., .AND., .OR., .EQV., .NEQV.
- 文字列の結合演算子 //
  - 'For'//'tran'
- 代入記号 =
  - X = 5 * Y + Z

選択文

- 論理 IF

```
IF ((N .EQ. 0) .OR. (N .EQ. 1)) THEN
   …
ELSEIF (N .GT. 3) THEN
   …
ELSE
   …
ENDIF
```

```
   IF (.NOT. ((N .EQ. 0) .OR. (N .EQ. 1))) GOTO 10
                …
   GOTO 30
10 IF (.NOT.(N .GT. 3)) GOTO 20
                …
   GOTO 30
20 CONTINUE
                …
30 CONTINUE
```

繰り返し文

- for-loop のみ!!!

```
   DO 10 I = 9, 1, -2
      …
10 CONTINUE
```

- 増分 は1であれば省略可能

無条件ジャンプと繰り返し

- 他の種類のループは 無条件ジャンプ で
  - GOTO 文を IF とともに用いる

```
10 IF (N .LE. 100) THEN
   …
   GOTO 10
   ENDIF
```

```
10 IF (N .GT. 100) GOTO 20
   …
   GOTO 10
20 CONTINUE
```

- 上記は while ループ

配列

- 宣言
  - REAL P(10)
  - INTEGER I(-5:5)
  - LOGICAL L(-1:2,3:5)
- 最大次元数 7. 添え字は整数
- 参照
  - I(-1) = 0
  - L(0, 4) = .TRUE.

サブルーチン

- 構文

SUBROUTINE *手続き名* (引数のリスト)
引数の型宣言
…
RETURN
END

- 引数リストはコンマで区切る
  - 引数はパラメータとも呼ぶ
- 引数の型宣言は一番最初
- 複数の RETURN が可能
- サブルーチン呼び出しは CALL 文による

サブルーチン

- 配列が引数のとき
  - 最後の次元を除いて宣言

```
SUBROUTINE TEXT(M, N, A)
INTEGER M, N
INTEGER A(M, N, *)
```

- 参照渡し
- 再帰はなし
- 関数も同様. 調べてみて下さい!

サブルーチン

- 例

```
M = 10
N = 20
CALL SWAP(M, N)
…
SUBROUTINE SWAP(X, Y)
INTEGER X, Y, T
T = X
X = Y
Y = T
RETURN
END
```

入出力

- パンチカードまたは紙テープ（知らないでしょう）
- 磁気テープ, 磁気ドラム, 磁気ディスク
- 入出力ユニット
- 単純な機能のみ提供
- *新規(?)* 機能
  - Direct-access
  - 内部ファイル
  - 未format
  - レコード




I/O – ファイルのオープンとクローズ

- ファイルのオープン
  - `OPEN(`*`unit`*`, FILE=`*`fname`*`, STATUS=`*`st`*`)`
  - unit: 1 から 99 までの任意の整数, 5, 6 を除く (理由は read/write文の説明を読めばわかる. もっともなぜ、5 と 6 なのだろう？)
  - status: `'new'`, `'old'`, `'unknown'`
- ファイルのクローズ
  - `CLOSE(`*`unit`*`)`

I/O – 読み込み

- 一行を読む:

```
      READ(unit, label) varname, …
label FORMAT(specifications)
```

- `unit = *` は標準入力（stdin）. これはずっと後の話
- 書式の仕様
  - I5: 5 桁の整数
  - F5.2: 浮動小数点数, 幅 5桁, 小数点以下 2 桁
  - A10: 文字列, 長さ 10 桁
  - 上記の数字はいずれも optional.

I/O – 読み込み

- 例

```
    READ(2, 10) J
    READ(2, 20) K, A
    READ(2, 30) C, D, E, F
10  FORMAT(I)
20  FORMAT(I5, F5.3)
30  FORMAT(2(I5, A))
```

I/O – 書き出し

- 一行書き出すには:

```
      WRITE(unit, label) varname, …
label FORMAT(specifications)
```

- `unit = *` は標準出力（stdout. ずっと後の話）
- 書式仕様は読み込みと類似.

プログラム01： Hello World

プログラム本体

```
c
c 01hello.f
c
      write(6,*) 'Hello World. Here is a Fortran'
      stop
      end
```

a.exe

コンパイルの仕方(cygwin の場合)

```
f77 01hello.f ← コンパイルをしている。これで a.out ができる
./a.exe       ← 実行している
```

以下堀 正岳氏のスライドを利用した

Fortran77 の約束事

横幅は72文字以内

```
c
c 01hello.f
c
      write(6,*) 'Hello World. Here is a Fortran'
      stop
      end
```

一列目に文字があったら、コメント行とみなす

プログラム文は7列目から書く

プログラムの終わりは stop, end の2行

write 文

画面や、ファイルに結果を書き込む命令

```
write(6,*) 'Hello World. Here is a Fortran'
```

出力先
6 は画面出力

書式なし、という意味

プログラム02： 四則演算　　02simplemath.f

```
c
c
c 02simplemath.f
c
c author: M.E.H.
c date:   2002,11/5
c
      pi = 3.141592      ! pi
      r  = 6378136       ! radius of earth

      rkm  = r / 1000
      area = 4 * pi * rkm**2
      area = area / 1000000

      write(6,*) 'area is ', area

      stop
      end
```

円周率と地球半径をメートルで設定

km に変換してから面積を計算。面積は 100万平方km に変換

プログラムを体積計算に変えられますか？

四則演算のいろいろ

$a \times \dfrac{b+c}{d}$

```
a * (b + c) / d
```

$\dfrac{(a+b) \times h}{2}$

```
(a + b) * h / 2
0.5 * h * (a + b)
```

$\dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

```
(-1 * b + sqrt(b**2 - 4*a*c)) / (2*a)
```

この括弧は必須！
なぜだか考えてみよう

プログラム03： 数学関数　　03mathfunc.f

```
c
c 03mathfunc.f
c
c
      pi  = 3.141592
      deg = 30
      rad = deg * pi / 180.0

      write(6,*) 'sin (',rad,' ) =', sin(rad)
      write(6,*) 'cos (',rad,' ) =', cos(rad)

      stop
      end
```

度をラジアンに変換

正弦と余弦を計算して出力

プログラムを体積計算に変えられますか？

数学関数のいろいろ

abs	絶対値	log	自然対数
mod	余剰。あまりのこと	exp	指数
int	整数部分 3.14 なら、3 を返す。		
sqrt	平方根。当然、負を与えてはいけない。		

tan	正接
asin	arcsin
acos	arccos
atan	arctan
sinh	双曲線正弦
cosh	双曲線余弦
tanh	双曲線正接

プログラム04： ループ　　04doloop.f

```
c
c 04doloop.f
c
c author: M.E.H.
c date:   2002,11/5
c
      do 10 i = 1,10
        write(6,*) 'counting up ...', i
10    continue

      do i = 10,1,-1
        write(6,*) 'counting down ...', i
      end do

      stop
      end
```

1 から 10 まで数え上げるループ

10 から 1 まで –1 の間隔で数えるループ

奇数だけのループを作れますか？

プログラム05： 2重ループ　　05multiloop.f

```
c
c 05multiloop.f
c
c author: M.E.H.
c date:   2002,11/5
c
      do i = 1,5
        do j = 1,5
          write(6,*) '(i,j) = ', i,j
      end do
      end do

      stop
      end
```

外側のループ

内側のループ

1. まず、i = 1 のまま、内側のループが j = 1,2,3,4,5 の5回、回ります
2. 次に i = 2 について、やはり内側のループが5回、回ります。

3重ループを作れますか？

プログラム06： if 分岐　　06ifthen.f

```
c
c 06ifthen.f
c
c author: M.E.H.
c date:   2002,11/5
c
      do i = 1, 100
        j = mod(i,7)
        if (j.eq.0) then
          write(6,*) 'multiples of seven',i
        end if
      end do

      stop
      end
```

i を 7 で割った余りを、j に代入

j が 0 なら、write する

二重 if を使って、7 と 5 の公倍数を出せますか？

プログラム07： if / else分岐　　07ifthenelse.f

```
c
c 07ifthenelse.f
c
c author: M.E.H.
c date:   2002,11/5
c
      do i = 1, 10
        j = mod(i,2)
        if (j.eq.0) then
          write(6,*) 'even number',i
        else
          write(6,*) 'odd  number',i
        end if
      end do

      stop
      end
```

j が 0 なら、even、そうでないなら、odd と表示する。

二重 if を使って、7 と 5 の公倍数を出せますか？

if で使える比較のいろいろ

(a.eq.b)	a は b に等しい （a equals b)
(a.ne.b)	a は b に等しくない （a not equal b)
(a.lt.b)	a < b　(a is less than b)
(a.le.b)	a <= b　(a is less than or equal to b)
(a.gt.b)	a > b　(a is greater than b)
(a.ge.b)	a >= b　(a is greater than or equal to b)
(a.and.b)	a かつ b
(a.or.b)	a もしくは b

プログラム08： read文　　08read.f

```
c
c 08read.f
c
c author: M.E.H.
c date:   2002,11/5
c
      pi = 3.141592

      write(6,*) 'input radius:'
      read(5,*) r

      area = pi * r * r

      write(6,*) 'area is',area

      stop
      end
```

キーボードからの入力を受け取り、r に代入する。

二つの値を受け取って、三角形の面積を求めさせるには？

プログラム09： 全部使った　09sample.f 前半

値を入力すると、それが素数かどうかを判別するプログラム

```
c
c 09sample.f
c
c author: M.E.H.
c date:   2002,11/5
c
      write(6,*) 'input an positive integer'
      read(5,*) i

      if ((i.le.0).or.(i.ge.60000)) then
        write(6,*) 'invalid input'
        stop
      end if
```

キーボードからの入力を受け取り、代入する。

値が負だったり、60000以上の数字ならエラーを画面に書いて、中断

プログラム09： 全部使った　09sample.f 後半

```
      iflag = 0

      do j = 2, i-1
        a = mod(i,j)
        if (a.eq.0) then
          iflag = 1
        end if
      end do

      if (iflag.eq.0) then
        write(6,*) i,'is a prime'
      else
        write(6,*) i,'is NOT a prime'
      end if

      stop
      end
```

iflag を 0 にセット

2 から、i - 1 までループ

もし、i が約数をもっていたら、iflag を 1 にセット

iflag が 1 なら、少なくとも一つの約数があったわけで、素数ではない

このプログラムは不完全です。見落としている数はどれ？

プログラム10： 型の実験　10types.f

```
      integer i,j
      real    a,b,c

      i = 3.1415
      j = 5 / 3

      a = 3.1415
      b = 5 / 3
      c = 5.0 / 3.0

      write(6,*) 'these are integers'
      write(6,*) 'i -->', i
      write(6,*) 'j -->', j
      write(6,*) 'these are real'
      write(6,*) 'a -->', a
      write(6,*) 'b -->', b
      write(6,*) 'c -->', c
      stop
      end
```

これらはinteger（整数）です
i --> 3
j --> 1

しかしこれはreal（浮動小数点数）
a --> 3.14150
b --> 1.00000
c --> 1.66667

出力

5 と 5.0ではコンピュータにとっては意味が違う！

処理系によってこの挙動は違います。注意！

変数の型について

変数には、「実数」、「整数」、「文字」などといった型がある

整数型の変数には、整数しか入れられない

```
integer  inum
inum = 3.14
```

→ エラーにはならないけど小数点以下は捨てられる

2 と 2.0 は違う。実数計算からは実数しか生まれない

```
real r
integer i
r = 6 / 2    → 3.0
i = 6 / 2    → 3
```

データの計算は実数で、ループ変数は整数で、と使い分ける

暗黙の型宣言について

特に宣言をしなくても

a-h, o-z で始まる変数は実数

i-n で始まる変数は整数

inum	整数
itime	整数
rt	実数
rvalue	実数

でも、わかりにくくて仕方ないので、一定のルールを導入するほうが賢い。

```
do i = 1,n
  do j = 1,m
    rdata = rdata * 0.98
  end do
end do
```

鉄則：

1. ループ変数は、i, j, k, itime, imonth といったものだけ
2. 絶対に「l、o」（L と O の小文字）を変数の最初に使わない

暗黙の型宣言に頼りすぎると、デバッグに不必要な苦労が…

プログラム11： 大きすぎる数　11bignum.f

```
c
c 11bignum.f
c
c author: M.E.H.
c date:   2002,11/5
c
      integer*2 inum
      real    rnum, rnum2

      inum = 32767
      write(6,*) 'inum = ', inum

      inum = inum + 1
      write(6,*) 'inum = ', inum

      stop
      end
```

inum = 32767

inum = -32768

出力

inum を 1 増やしたのに、負になってしまった！

大きすぎる数はエラーのもと！

変数が、正確に保持できる数字の大きさには限界がある。

特に整数は 65536 あたりで負に転ずることもあるので、注意が必要。実数はあまり気にする必要はない。

単精度	31+1bit	1.17×10^{-38} ～ 3.40×10^{38}
倍精度	63+1bit	2.22×10^{-308} ～ 1.79×10^{308}

プログラム12： 小さすぎる数　12roundoff.f

```
c
c 12roundoff.f
c
c author: M.E.H.
c date:   2002,11/5
c
      real   r
      real   s1
      double precision s2

      do r = 1,10000
        s1 = s1 + 1/(r**2)
        s2 = s2 + 1/(r**2)
        write(6,*) r,s1,s2, s1-s2
      end do

      stop
      end
```

$$S = \sum_{i=1}^{n} \frac{1}{i^2}$$

を単精度と倍精度の両方で計算している

出力

4097 あたりから、単精度の結果はかわらない！
つまり、小さすぎる数も表現できていない
→「桁落ち誤差」

以上 堀 正岳氏のスライドを利用した

```
C
C 51HELLO.F
C
      WRITE(6,600)
 600  FORMAT(30HHELLO WORLD. HERE IS A FORTRAN)
      STOP
      END
```

```
C
C 52SIMPLEMATH.F
C
C
C  PI AND RADIUS OF EARTH
      PI = 3.141592
      R  = 6378136

      RKM  = R / 1000
      AREA = 4 * PI * RKM**2
      AREA = AREA / 1000000

      WRITE(6,600) AREA
 600  FORMAT(8HAREA IS , F15.7)
      STOP
      END
```

```
C
C 54DOLOOP.F
C

      DO 10 I = 1,10
        WRITE(6,600)  I
 10   CONTINUE

      DO 20 I = 10,1,-1
 20   WRITE(6,610)  I

 600  FORMAT(15HCOUNTING UP ..., I3)
 610  FORMAT(17HCOUNTING DOWN ..., I3)

      STOP
      END
```

```
C
C 53MATHFUNC.F
C
      PI  = 3.141592
      DEG = 30
      RAD = DEG * PI / 180.0

      WRITE(6,600) RAD, SIN(RAD)
      WRITE(6,600) RAD, COS(RAD)
 600  FORMAT( 5HSIN (, F15.7,4H ) =, F15.7 )

      STOP
      END
```

```
C
C 55MULTILOOP.F
C
      DO 50 I = 1,3
        DO 50 J = 1,3
          WRITE(6,600)  I,J
 50   CONTINUE

C FOLLOWING NEW LINE (OR NEW PAGE 1H1,
C OVERWRITE 1H+) DOES NOT WORK
C
      WRITE(6,610)
 610  FORMAT(1H )

      DO 60 I = 1,3
        DO 60 J = 1,3
          WRITE(6,600)  I,J
 60   CONTINUE

      STOP
 600  FORMAT(8H(I,J) = , 2I4 )
      END
```

```
C
C 56IFTHEN.F
C
C

      DO 10 I = 1, 100
        J = MOD(I,7)
 10     IF (J.EQ.0) WRITE(6,600) I

      STOP
 600  FORMAT(18HMULTIPLES OF SEVEN, I4)
      END
```

```
C
C 57IFTHENELSE.F
C
C
      DO 10 I = 1, 10
        IF (MOD(I,2).EQ.0) GOTO 20
C BRANCH TO 'ELSE' PART
        WRITE(6,610) I
        GOTO 30

C OR BRANCH TO 'THEN' PART
 20     WRITE(6,600) I

C AND THEN COME TOGETHER
 30   CONTINUE
 10   CONTINUE

      STOP
 600  FORMAT(11HEVEN NUMBER, I4)
 610  FORMAT(11HODD  NUMBER, I4)
      END
```

```
C
C 58READ.F
C
C
      PI = 3.141592

      WRITE(6,600)
 600  FORMAT(13HINPUT RADIUS:)
      READ(5,500) R
 500  FORMAT(E6.3)

      AREA = PI * R * R

      WRITE(6,610) R
 610  FORMAT(9HRADIUS IS, E15.7)
      WRITE(6,620) AREA
 620  FORMAT(9HAREA   IS, E15.7)

      STOP
      END
```

```
C
C 59SAMPLE.F
C
      WRITE(6,600)
 600  FORMAT(37HINPUT AN POSITIVE INTEGER IN 5 DIGITS)
      READ(5,500) I
 500  FORMAT(I5)

      IF ((I.GT.0).AND.(I.LE.99999)) GOTO 10
        WRITE(6,610)
 610    FORMAT(13HINVALID INPUT)
        STOP
 10   CONTINUE

      IFLAG = 0
      JMAX = SQRT(FLOAT(I))
      DO 20 J = 2, JMAX, 2
        IF (MOD(I,J).NE.0) GOTO 20
          IFLAG = 1
          GOTO 30
 20   CONTINUE
 30   CONTINUE

      IF (IFLAG.EQ.0) WRITE(6,620) I
      IF (IFLAG.EQ.1) WRITE(6,630) I
 620  FORMAT(I5, 11H IS A PRIME)
 630  FORMAT(I5, 15H IS NOT A PRIME)

      STOP
      END
```

Fortran コンパイラ

フリーのものがいくつかあります。

- GFortran (cygwin, MinGW, Mac OS, Linux)
  - http://gcc.gnu.org/wiki/GFortranBinaries
- 「フリーのFortran95」
  - http://plaza.rakuten.co.jp/takaamahara/diary/200805170000/
  - http://d.hatena.ne.jp/arakik10/20120214/1329167074
- WATCOM Fortran
  - http://www2.nc-toyama.ac.jp/~mkawai/almanac/nadown/watcom/watcom.html
- Intel Fortan Compiler for Mac OS
  - http://www.xlsoft.com/jp/products/intel/compilers/fcm/

肝心のコンパイラへのリンク先が切れている

GFortran

- Google で検索