数値計算法

第四回
FORTRANによるプログラミング
＆
プログラミングの例

若狭　智嗣

粒子物理学講座

circle.fの解説

左は6文字空いている

1行目はプログラム名を宣言

全ての変数は定義する事を宣言

```
      program circle
      implicit none
c constant:
      real PI                          ! 円周率PIの値を入れる変数
      parameter (PI=3.141593)          ! 円周率PIを定義
c local variables:
      real radius                      ! 半径の値を入れる変数
      real area,volume                 ! 面積、体積の値を入れる変数
c begin:
      radius = 10.0                    ! 半径として10.0を入れる
      area   = PI * radius**2          ! *は乗算の、**はべき乗の記号
      volume = 4.0 * PI * radius**3 / 3.0 ! /は除算の記号
      write(*,*) 'radius = ',radius    ! 半径を画面に表示
      write(*,*) 'area   = ',area      ! 面積を画面に表示
      write(*,*) 'volume = ',volume    ! 体積を画面に表示
      stop
      end
```

1文字目がcの場合、その行はコメントとみなす

文字列はクオート(‘)で括る。表示は、
volume = 体積値
となる

circle.fの問題点（改良したい点）

- circle.fは、円の面積などを１回しか計算しない。
- いちいちプログラム中の半径の値を変更して、コンパイルした後、実行するのは手間である。
  - 半径を、ある刻み幅で変更するなら、ループ文
    do…end do
    や、
    do while()…end do
    が使える
  - 面積や体積を計算する部分は、他のプログラムでも使えるので、部品化したい
    →サブルーチンや関数にする
  - 半径をプログラムに埋め込むのではなく、後でキーボードから入力したい
    →read(*,*)を使う

circle2.fの解説
–繰り返しとサブルーチン–

mule@ah

ptions Buffers Tools Fortran Help

サブルーチン
(下請けプログラム)
calcを呼ぶ(callする)

半径がRADIUS_INIからRADIUS_MAXまで
RADIUS_DELTA刻みで計算させる

```
      program circle2
      implicit none
c constants:
      real RADIUS_INI,RADIUS_MAX                  ! 半径の初期値と最大値
      parameter(RADIUS_INI=1.0,RADIUS_MAX=10.0)
      real RADIUS_DELTA                           ! 半径の刻み幅
      parameter(RADIUS_DELTA=2.0)
c local variables:
      real radius                                 ! 半径の値を入れる変数
      real area,volume                            ! 面積、体積の値を入れる変数
c begin:
      radius = RADIUS_INI                         ! 半径を初期化
      do while (radius.le.RADIUS_MAX)             ! 半径が最大値になるまで繰り返し
         call calc(radius,area,volume)            ! サブルーチンで面積・体積計算
         write(*,*) 'radius = ',radius
         write(*,*) 'area   = ',area
         write(*,*) 'volume = ',volume
         write(*,*)
         radius = radius + RADIUS_DELTA
      end do
      stop
      end
```

カッコ内の条件が満たされるまで、挟まれた内容を実行する

calcにradiusを渡し、areaとvolumeを返してもらう

radiusをRADIUS_DELTAだけ増やす

calc.fの解説
ーサブルーチンcalcの説明ー

1行目はサブルーチン名を宣言

サブルーチンを呼ぶプログラムとの間で受け渡しする変数を列挙

File Edit Options Buffers

```
      subroutine calc(radius,area,volume)
      implicit none
c constant:
      real PI                                  ! 円周率PIの値を入れる変数
      parameter (PI=3.141593)                  ! 円周率PIを定義
c input:
      real radius                              ! 半径の値を入れる変数
c outputs:
      real area,volume                         ! 面積、体積の値を入れる変数
c begin:
      area   = PI * radius**2                  ! *は乗算の、**はべき乗の記号
      volume = 4.0 * PI * radius**3 / 3.0      ! /は除算の記号
      return                                   ! 呼んだ上位のプログラムに戻る
      end
```

列挙した変数も型宣言する

サブルーチンを呼んだプログラムに戻る（returnする）

サブルーチンの終わり

実習

- メインプログラムは
  circle2.f
  として、サブルーチンは、
  calc.f
  として置いてある。各人のディレクトリにコピーせよ。
- 複数のプログラムから、実行形式（circle2とする）を生成するには、

スペース

```
% frt -o circle2 circle2.f calc.f
```

プログラムファイルをスペースで区切って列挙

とすればよい。各人実行せよ。
- サブルーチンで、円の周りの長さ and/or 球の表面積を計算して、結果を表示するようにプログラムを変更してみよ

circle3.fの解説
ー端末からの読み込みの例ー

mule@ah

File Edit Options Buffers Tools Fortran Help

```
      program circle3
      implicit none
c local variables:
      real radius                          ! 半径の値を入れる変数
      real area,volume                     ! 面積、体積の値を入れる変数
c begin:
      write(*,*) 'radius? '                ! 画面にradius?と表示
      read(*,*) radius                     ! 半径を読み込む
      call calc(radius,area,volume)        ! サブルーチンで面積・体積計算
      write(*,*) 'radius = ',radius        ! 半径を画面に表示
      write(*,*) 'area   = ',area          ! 面積を画面に表示
      write(*,*) 'volume = ',volume        ! 体積を画面に表示
      stop
      end
```

read(*, *)

標準入力(キーボード)から入力

書式を計算機に任せる
(複数の値を読み込むとき、各々はタブまたは1つ以上のスペースで区切られていればよい)

実習

- circle3.fをコンパイルし、実行しなさい。
- このプログラムでは、radius? と表示した後で改行してしまう。
- 改行させないためには、write文を以下の何れかのように書き改めればよい。
  - write(*,’(“radius? ”,$)’)
  - write(*,’(a,$)’) ‘radius? ’
- 実際に書き換えて、改行しないこと（＝radius? の後に半径を入力できる）ことを確認せよ。

プログラミングの例

- データ(数表)が与えられた時に、そのデータの**平均値**と各データの平均値からの**ズレ(偏差)**を計算・表示するプログラムを作る。
- プログラムとデータは全て、以下のディレクトリにある。
  /home/teacher/z6wt01in/SAMPLE/
- データファイル（weight.dat）の中身は以下の通り

mule@ah

File Edit Options Buffers Tools Help

```
0.735
0.886
0.781
0.927
0.670
0.579
0.511
0.277
0.982
0.165
0.950
0.305
```

メインプログラム（mean.f）の流れ

- **メインプログラムでは、以下の処理を行う**
  - データファイルを開き
  - データファイルからデータを読み込み、配列に格納する
  - 配列に格納したデータを使って、平均値を計算する
  - 先ほど計算した平均値を使い、各データの偏差を計算する
  - 結果を出力する
- **各処理は、サブルーチンまたは関数に下請けさせる。**
  - 下請けのプログラムは、部品として使いまわしができる。

mean.fの説明

mule@ah

File Edit Options Buffers Tools Fortran Help

```
      program calc_mean_value
      implicit none
c constants:
      integer MAX_DATA                          ! 読み込むデータ数の最大値
      parameter(MAX_DATA=100)                   ! 100に定義
      integer ERR                               ! ファイル操作の状態を表す
      parameter(ERR=-1)                         ! 問題有り:-1(無し:0)
      integer LUNIN                             ! 入力ファイルの論理機番
      parameter(LUNIN=11)                       ! 11を定義
c function:                                     ! 組み込み関数以外は宣言する
      real calc_mean                            ! 平均値を計算するユーザー関数
c local variables:
      real    data_array(max_data)              ! データを格納する配列
      integer status                            ! ファイル操作の状態
      integer num_data/0/                       ! データの数
      real    mean/0.0/                         ! 平均値
c begin:
```

mean.fの説明（続き）

```
c begin:
 1      continue
        call file_open('input',LUNIN,status)
        if (status.eq.ERR) goto 1
        call read_data(LUNIN,MAX_DATA,num_data,data_array)
        if (num_data.gt.0) then
           mean = calc_mean(num_data,data_array)
           call data_output(num_data,data_array,mean)
        else
           write(*,*) 'data empty'
        endif
        stop
        end
```

ファイルに、論理機番LUNINを割り当てて開く

論理機番LUNINからデータを読み、配列data_arrayに格納する。データの数はnum_dataに格納する。

平均値を計算(calc_mean)し、結果をmeanに代入（=）する。
平均値と、平均値と各データのズレ（偏差）を、サブルーチン:data_outputを使って出力する。

file_open.fの説明

- プログラムの中身を理解する必要は必ずしも無い
  - 部品（道具）として扱えばよい
- 但し、正しく扱うには機能（引数の意味）を正しく理解する必要がある。
- このプログラムの引数は３つ
  - 第一引数：mode（文字列型）
    - ‘input’か’output’のいづれか
    - データを読むときは’input’を、書き込むときは’output’を指定
  - 第二引数:lun（整数型）
    - ファイルに割り当てる論理機番を指定（１－９９）
    - ０，５，６，７は特別な意味があるので使用しないこと
  - 第三引数:status(整数型)
    - ファイルが正しく開けたか否か（引数が正しいか）
    - 正しければNORMAL(=0)、問題があればERR(=-1)

read_data.fの説明

無限ループでデータを読み込む

```
c begin:
      num_data = 0                                  ! データ数を初期化
      do while (FOREVER)
         read(lunin,*,end=999) value                !
         num_data = num_data + 1
         array(num_data) = value
         if (num_data.gt.size) then
            write(*,*) 'Number of data exce
            to 999
         end
      end do
999   contin
      return
      end
```

Lunin（ファイル）からデータを読み込みvalueに格納する。
ファイルの最後に行き着いたら、行番号(1文字目から5文字目)999に飛ぶ。

データ数(num_data)を1つ増やす。
配列data_arrayのnum_data番目にvalueを格納（代入）する

calc_mean.fの説明

mule@ah

File Edit Options Buffers Tools Fortran Help

```
      real function calc_mean(num_data,array)
      implicit none
c inputs:
      integer num_data                    ! データの数
      real    array(num_data)             ! データが入った配列
c local variable:
      integer i                           ! ループ変数
      real    sum                         ! データの総和
c begin:
      sum = 0.0                           0に初期化
      do i=1,num_data
         sum = sum + array(i)             ! データの総和を計算
      end do
      calc_mean = sum / float(num_data)   ! 総和をデータ数で除算
      return
      end
```

$$\mathbf{sum} = \sum_{i=1}^{\mathbf{num_data}} \mathbf{array}(i)$$

data_output.fの説明

mule@ah

File Edit Options Buffers Tools Fortran Help

```
      subroutine data_output(num_data,array,mean)
      implicit none
c inputs:
      integer num_data                          ! データの数
      real    array(num_data)                   ! データが入った配列
      real    mean                              ! 平均値
c local variable:
      integer i                                 ! ループ変数
c begin:
      do i=1,num_data
         write(*,'(I3,A1,2F9.4)')
     &        i,':',array(i),array(i)-mean
      end do
      return
      end
```

i番目のデータ
array(i)
と、i番目のデータの偏差
array(i)-mean
を表示（meanは平均値）

課題（標準偏差と偏差値の計算）

- 54ページの練習を第一回レポート問題とする。

練習

本章で解説したプログラムを改良し、平均値と標準偏差を出力するプログラムを作成せよ。また、それぞれのデータの偏差値 (平均が 50、標準偏差が 10 になるように変換した値) を出力するプログラムを作成せよ。

平均値 $$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x(i)$$

分散 $$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}\left(x(i) - \bar{x}\right)^2$$

標準偏差 $$\sigma = \sqrt{\sigma^2}$$

レポート提出に際して

- **各人がプログラムした部分（または修正した部分）のプログラムを添付すること。**
  - サンプルとして与えたプログラム（サブルーチンや関数）は添付しなくてよい（紙は無駄にしない）
  - プログラムの印刷の仕方は講義ノートの21ページを参照のこと
- **プログラムの出力結果を添付すること**
  - 平均値の値は必須
  - 偏差値については、必ずしも全てのデータについて載せる必要はない（最初の5つ程度だけでもよい）
  - リダイレクションを使えば結果をファイルに書き出せるので簡単に添付できる（画面を書き取ってもよいが）
- **提出は次回講義終了時**

  11月6日、正午

  とする。

偏差値

- 偏差値yは、データxを以下の条件を満たすように変換した値
  - 平均値: $\bar{y} = 50$　　標準偏差: $\sqrt{\sigma_y^2} = 10$
- 線形変換　$y = ax + b$　を考える

$$\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y(i) = \frac{1}{N}\sum_{i=1}^{N}\{ax(i)+b\}$$

$$= a\frac{1}{N}\sum_{i=1}^{N} x(i) + b = a\bar{x} + b = 50$$

$$\sigma_y^2 = \frac{1}{N}\sum_{i=1}^{N}\left(y(i)-\bar{y}\right)^2 = \frac{1}{N}\sum_{i=1}^{N}(ax(i)-a\bar{x})^2$$

$$= a^2\frac{1}{N}\sum_{i=1}^{N}\left(x(i)-\bar{x}\right)^2 = a^2\sigma_x^2 = 100$$

$$a = \frac{10}{\sigma_x},\ b = 50 - \frac{10\bar{x}}{\sigma_x} \longrightarrow y = \frac{10(x-\bar{x})}{\sigma_x} + 50$$