計算物理学II (第8回)

今回の内容

- ファイル入出力
- グラフ作成

標準入出力

- 標準入力(キーボードからの入力)

```
read (*,*) var
read (5,*) var1, var2
```

```
scanf(“%d”, &var);
scanf(“%lf %lf”, &var1, &var2);
fscanf(stdin,”%d”, &var);
```

- 標準出力(画面への出力)

```
write (*,*) var
write (6,*) “var1 var2 = “, var1, var2
print *,  var
```

```
printf(“%d¥n”, var);
printf(”var1 = %f,   var2 = %f¥n”, var1,var2);
fprintf(stdout,”%d¥n”, var);
```

- 標準エラー出力(画面への出力)

```
write (0,*) “Error”
```

```
fprintf(stderr, “Error¥n”);
```

Fortran

read, writeの１つ目の引数(unit)で**ファイル装置番号**を指定する。*とすると標準入出力になるが、5が標準入力、6が標準出力、0が標準エラー出力となっている

C

入出力の型を明示。 %d 整数型、%f 実数型 %lf 倍精度実数型(入力のみ指定)
scanfのときは読み込みたい変数の前に&をつける(変数のアドレス)。 ¥nは改行
stdin(標準入力), stdout(標準出力), stderr(標準エラー出力)はstdio.hで定義されている

ファイル入出力(リダイレクト)

シェル側でファイルに入出力

シェルでリダイレクトによって標準入出力をファイルに変更
Linuxのリダイレクトの部分を復習してください

実行ファイルa.outを実行するときの標準入力を inputという名前のファイル、
標準出力を output 標準エラー出力をerrorという名前のファイルにする場合は

```
./a.out < input > output  2> error
```

標準出力と標準エラー出力両方をoutputに書き出す場合

```
./a.out > output  2>1&
```

プログラム内でのファイル入出力

Fortranでファイルを指定

```
! 装置番号1としてinputfile.txtを開く
open( 1, file='inputfile.txt', action='read')
! 装置番号2としてoutputfile.txtを開く
open( 2, file='outputfile.txt', action='write')
! 装置番号1(inputfile.txt)から読み込んだ値をvarに代入
read(1,*) var
! 装置番号2(outputfile.txt)に変数varの値を出力
write(2,*) var

close(1) ! 装置番号1のファイルを閉じる
!(この後装置番号1は他のファイルに当てることができる)
close(2) ! 装置番号2のファイルを閉じる
```

Cでファイルを指定

```
FILE *fp1, *fp2;
/* FILE型ストリーム。stdio.hで定義されている */
fp1=fopen('inputfile.txt', 'r');
 // inputfile.txtを読み込み専用として開く
fp2= fopen('outputfile.txt', 'w');
// outputfile.txtを書き込み専用として開く。

fscanf(fp1, "%lf", &var);
// fp1ストリームから倍精度実数varを読み込む
fprintf(fp2, "%f", var);
// fp2ストリームに倍精度実数varを書き込む
fclose(fp1);  // ファイルを閉じる
fclose(fp2);  // ファイルを閉じる
```

Fortran: 装置番号5,6,0は標準入力、出力、エラー出力に予約されているのでこれ以外の番号を指定
C: fopen関数の読み書きのモード：
r 読み込み, w 書き込み, **a 追加書き込み**(ファイルが存在する場合は末尾から書き込む)

ファイル入出力(Fortran)

open関数で指定できるのは

unit指定子:	装置番号を指定。 unit=は省略して数字だけでもOK
file指定子：	ファイル名を指定
status指定子：	‘old’ すでにファイルが存在する場合
	‘new’ ファイルが新しく作られる場合
	‘replace’ ファイルが存在する場合は前のファイルが削除される
	‘unknown’ デフォルト値。ファイルがない場合は新しく作成。
action指定子：	‘read’ 読み込み専用でファイルを開く (間違えて書き込むミスを防ぐ)
	‘write’ 書き込み専用でファイルを開く
	‘readwrite’ デフォルト。読み書き可能
position指定子:	‘asis’ デフォルトの場所、通常ファイルの先頭位置
	‘rewind’ ファイルの先頭位置
	‘append’ ファイルの末尾位置

すでに存在するファイルに追記したい場合は

```
open(unit=1, file='ファイル名', status='old', position='append')
```

などとしてファイルを開く

書式指定(Fortran)

- write,printで出力の書式指定ができる
- writeの2つ目の引数、またはprintの*を書式に置き換える

```
write(*,'(i5)')   ivar          ！ 5桁までの整数を表示
write(*,'(3f15.8)')   x, y, z  ！ 実数を15桁で、小数点以下は8桁として3つ表示
write(*,'(2i5,3f15.8)') i, j, x, y, z  ！5桁までの整数を2つ表示、その後15桁,小数点以下8桁で実数を3つ表示
```

編集記述子	形式	意味
i	iw	幅wの整数
f	fw.d	幅w,小数点以下d桁の実数
e	ew.d	幅w,小数点以下d桁の実数
a	aw	幅wの文字列
x	x	空白
/	/	改行

```
write(*,'(3x,i5)') 123
write(*,'(f8.2)') 123.4567
write(*,'(e8.2)') 123.4567
write(*,'(a8)') 'abc'
```

```
_ _ _ _ _ 1 2 3
_ _ 1 2 3 . 4 5
_ 0 . 1 2 E + 3
_ _ _ _ _ A B C
```

書式指定しない(*と書く)場合はデフォルトの書式が適用される(システム依存)

書式指定(Fortran)

- 書式指定は文字列配列に代入したものを指定してもOK
- writeの指定子はunit(装置番号を指定)、fmt (書式を指定)
- 文字列操作で書式をプログラム実行中に決定したい場合に使える

```
real*8 :: a, b
character(50) :: format1

format1 = '(a, f15.5)'                          ! 文字列配列format1に書式を代入

write(unit=*, fmt=format1)  "a = ", a   ! 書式はformat1で指定する
write(unit=*, fmt=format1)  "b = ", b
```

書式指定(C)

- printf, fprintfで出力の書式指定ができる。

	サイズ指定なし	サイズ指定
整数	%d	%5d (5桁の整数)
実数	%f	%15.10f 全桁数15で小数点以下が10桁
実数(指数表示)	%e	%15.10e
文字	%c	
文字列	%s	%10s（10桁の文字列)

- scanf, fscanfでは単精度・倍精度実数も明示的に指定する

	サイズ指定なし
整数	%d
倍精度整数	%ld
実数	%f
倍精度実数	%lf

バイナリ入出力(発展)

- バイナリ入出力 (バイナリ=2進数)
- 倍精度実数は10進数の形式 (1.00000E+02など)で出力すると情報が落ちる。(メモリ上では64ビット、仮数部と指数部に分けて2進数で保持)
- ファイル入出力するときに2進数(バイナリ)のまま扱うことができる
- ただしテキストとして開いて読むことはできない(0と1が並んでいるだけでありバイナリエディタが必要だが読めたものではない)。
- 計算の途中経過出力、途中から再開するときなどに使う。図にする必要のないデータの保存もバイナリで行うのが一般的

バイナリファイルに出力する

```
open(1, file='output.dat', form='unformatted')
write(1) var  ! unformattedの場合は書式なし出力
close(1)
```

```
fp=fopen('output.dat', 'wb'); //wbはバイナリ書き込み
// varが倍精度実数の場合、
//fwrite(データアドレス、データ一つのサイズ、個数、ストリーム)
fwrite(&var,sizeof(double),1,fp);
fclose(fp);
```

バイナリファイルから読み込む

```
open(2,file='output.dat', form='unformatted')
read(2) var     ! 書式なしのread文
close(2)
```

```
fp=fopen('output.dat', 'rb');// rbはバイナリ読み込み
fread(&x, sizeof(double),1,fp);
//xに倍精度実数データを格納
fclose(fp);
```

グラフ作成

出力するときはgnuplotなどで読み込みやすい形式で

gnuplotでは一行ずつデータを読み込んで特定の列と特定の列をx, yとしてグラフに描画

y=f(x)を0から10までの領域で書きたい場合 (関数f(x)は用意されていると仮定)
f(x)の値だけではなくxの値も描画のためには必要

```
open(10,file="fx.dat")
x = 0.0d0
do
    write(10,*) x, f(x)
    x = x + 1.0d-2
    if( x > 10.0d0) exit
end do
close(10)
```

```
fp=fopen("fx,dat","w");
for( x = 0.0; x<=10.0;x+=0.01){
    fprintf(fp,"%f %f ¥n" , x, f(x));
}
fclose(fp);
```

刻み幅は0.01としているが図にするときになめらかに見える程度に選べば良い。
標準出力に出力してもいいが、プログラムの他の場所で標準出力を使うかもしれないので
グラフに必要なデータ以外と混在しないように独立のファイルに出すほうがよい。

演習

- ファイル入出力 (15)
  - 3x3の行列を2つのファイルから2つ読み込んで行列積を計算、結果を他のファイルに出力

- グラフ作成 (16,17)
  - (16) gnuplotで読み込めるデータファイルの作成
  - (17) ロジスティック写像