# 2023-12-15_14_PLAIN_BINNARY_2D_3D

/work09/am/2023_PROGRAM/2023-12-15_14_EXERCISE



# 目的

2次元と３次元の配列に記憶されたデータをプレーンバイナリ形式で読み書きする練習

# 参考資料

- バイナリファイルの書き出しと読み込み

  https://gitlab.com/infoaofd/lab/-/blob/master/FORTRAN/PROGRAM_2022/F_05_BINARY_FILE.md

- データ構造とバイナリファイルの入出力

  https://gitlab.com/infoaofd/lab/-/blob/master/FORTRAN/PROGRAM_2022/F_05_SP01_EX_BINARY_IO.md

- ダイレクトアクセスの概念

  https://gitlab.com/infoaofd/lab/-/blob/master/FORTRAN/PROGRAM_2022/DIRECT_ACCESS.pdf

- GrADSでのバイナリーデータの取り扱い

  https://gitlab.com/infoaofd/lab/-/blob/master/FORTRAN/PROGRAM_2022/BINARY_DATA_GRADS.pdf

# サンプルプログラム

BINDIR2D_OUT.f90

```
program BIN2D_OUT

real,allocatable,dimension(:,:)::a
character(len=100)::infle,ofle
integer,parameter::IM=9,JM=9

ofle="BINDIR2D.BIN"
!print *,infle

!配列の割付
allocate(a(IM,JM))

!レコード長の取得
!ここでは1行分のデータを1レコードとする
isize=IM*4
```

```
!配列要素の設定
do j=1,JM
do i=1,IM
a(i,j)=float(i)*float(j)
end do !i
end do !j


!バイナリファイル(ダイレクトアクセス)の書き出し
open(13,file=ofle,form="unformatted",access="direct",recl=isize)

do j=1,JM
! 配列のj行目のデータをj番目のレコードとして書き出す。
write(13,rec=j)(a(i,j),i=1,IM)

!(a(i,j),i=2,IM)は、jの値を固定してiだけ変化させるという意味
end do !j

close(13)

!念の為画面表示させて確認（必須ではないが今回初めて行う例なので念の為）
do j=1,JM
print '(100f5.0)', (a(i,j),i=1,IM)
end do !j

!書き込みファイル名の確認
print "(a,a)","OUTPUT FILE = ",trim(ofle)

end program
```

## 注意

ifortでコンパイルする場合-assume bytereclオプションを付けること。

```
ifort -assume byterecl BINDIR2D_OUT.f90 -o BINDIR2D_OUT.exe
```

下記の他のプログラムも同様。

理由: Intel Fortran コンパイラのダイレクトアクセスのレコード長 (open文で RECL= で設定する値) の単位は,
デフォルトでは「ワード(4バイト)」である。一般的な「バイト」単位でレコード長を指定するには, コンパイルオプション -assume byterecl を指定する必要がある。

BINDIR2D_READ.f90

```
program BIN2D_OUT

real,allocatable,dimension(:,:)::a
character(len=100)::infle,ofle
integer,parameter::IM=9,JM=9

infle="BINDIR2D.BIN"
!print *,infle
```

```
!配列の割付
allocate(a(IM,JM))

!レコード長の取得
!ここでは1行分のデータを1レコードとする
isize=IM*4

!バイナリファイル(ダイレクトアクセス)の書き出し
open(13,file=infle,form="unformatted",access="direct",recl=isize,action='read')

do j=1,JM
!j番目のレコードを配列のj行目のデータとして読み込む
read(13,rec=j)(a(i,j),i=1,IM)

!(a(i,j),i=2,IM)は、jの値を固定してiだけ変化させるという意味
end do !jj

close(13)

!念の為画面表示させて確認（必須ではないが今回初めて行う例なので念の為）
do j=1,JM
print '(100f5.0)', (a(i,j),i=1,IM)
end do !j

!読み込みファイル名の確認
print "(a,a)","INPUT FILE = ",trim(infle)

end program
```

BINDIR3D_OUT.f90

```
program BIN3D_OUT

real,allocatable,dimension(:,:,:)::a
character(len=100)::infle,ofle
integer,parameter::IM=9,JM=9,KM=2

ofle="BINDIR3D.BIN"
!print *,infle

!配列の割付
allocate(a(IM,JM,KM))

!レコード長の取得
!ここでは一つのkに対してレコード1個とする
isize=IM*JM*4
!IM*JMの2次元のデータを1レコードとする。

!配列要素の設定
do k=1,KM
do j=1,JM
```

```
do i=1,IM
a(i,j,k)=float(i)*float(j)*float(k)
end do !i
end do !j
end do !k


!バイナリファイル(ダイレクトアクセス)の書き出し
open(13,file=ofle,form="unformatted",access="direct",recl=isize)

do k=1,KM
! 配列のk層のデータをk番目のレコードとして書き出す。
write(13,rec=k)((a(i,j,k),i=1,IM),j=1,JM)

end do !k
close(13)



!念の為画面表示させて確認（必須ではないが今回初めて行う例なので念の為）
do k=1,KM
print '(A,i3)','k=',k
do j=1,JM
print '(100f5.0)',(a(i,j,k),i=1,IM)
end do !j
end do !k

!書き込みファイル名の確認
print "(a,a)","OUTPUT FILE = ",trim(ofle)

end
```

## 演習

1. 上記のプログラムを自分で打ち込んでみて、何も見ないで作成できるようにする。
2. BINDIR2D_READ.f90を参考に、BINDIR3D_OUT.f90で作成されたプレーンバイナリファイルを読み込むプログラム、BINDIR3D_READ.f90を作成し、データが正しく読み込まれているかどうか実行結果を確認する。