Sobre la lectura y escritura de datos en FORTRAN.

1. Las sentencias READ /WRITE

Uno de los elementos básicos de la programación e cualquier lenguaje es la lectura y/o escritura de datos. En el lenguaje FORTRAN la lectura escritura se realiza a través de las sentencias READ y WRITE

A la hora de leer/escribir datos necesitamos definir:

- ¿Qué tipo de datos son?
- ¿Dónde queremos leerlos/escribirlos?
- ¿Cómo queremos leerlos/escribirlos.

El esquema general es el de

WRITE(___,___) ____

Igualmente con la sentencia READ.

En el primer campo se especifica donde están (o deben ponerse) los datos (si ponemos un * se tomará la pantalla). En el segundo campo como deben escribirse, (si ponemos un * se tomará la opción por defecto) , y en el tercer campo la lista de variables a leer/escribir. Adicionalmente a las variables también podemos escribir texto explicativo, poniéndolo entre comillas simples.

WRITE(*,*)'Resultados = 'A,B

Hará aparecer en pantalla el texto

Resultados=

Seguido de los valores de las variables A y B.

2. Sobre el tipo de datos

En referencia al tipo de datos que son distinguiremos:

- Que tipo de variables constituyen, es decir si son almacenados en una variable entera, real, doble precisión, compleja, etc,
- Cómo es la dimensión de dicha variable, es decir si es un vector, matriz, etc, y de que tamaño máximo.

Ambas funciones las realizamos con las mismas sentencias, para lo cual, al principio del programa podemos utilizar las siguientes sentencias

REAL	Variables reales en precisión simple
INTEGER	Variables enteras
DOUBLE PRECISION	Variables reales en precisión doble
COMPLEX	Variables complejas
CHARACTER	Variables alfanumericas (texto)

Recordemos que las variables reales o enteras aparecen descritas por defecto con el criterio de la primera letra, es decir aquellas variables que comienzanpor una letra de la A-H O-Z son reales y las que comienzan por las letras I;J,K,L,M,N son enteras.

En el caso de vectores o matrices debemos especificar el valor máximo de la dimensión del mismo, por ejemplo

```
REAL X(100),AM(50,10)
```

indica un vector real de hasta 100 elementos y una matriz de 50x10 elementos.

3. Sobre donde escribir/leer los datos

En cuanto al lugar donde queremos escribir/leer los datos está indicado por el primer elemento entre paréntesis de la sentencia READ/WRITE. Un * indica la pantalla.

```
READ(*,*) A, B
```

En el caso de que queramos leer/escribir en un fichero es necesario definir previamente dicho fichero. Para ello utilizamos la sentencia OPEN

La sentencia OPEN permite abrir un fichero, al cual necesitamos asignarle un número o unidad para que el ordenador lo identifique aparte del nombre y del estatus que indica si el fichero es preexistente o si debe ser creado por el programa. Continene los siguientes elementos

- **Unit:** suministrado por el programador. Indica el número con el que identificamos el fichero, luego ese número será utilizado en las sentencias READ o WRITE.
- **File**: Suministrado por el programador, indica el nombre del fichero. Debe ir entre comillas simples ya que es un texto. Si no ponemos un directorio específico el ordenador buscará o creará el fichero en el directorio que este asumiendo por defecto en el ordenador.
- **Status**: Suministrado por el usuario puede ser: ‘OLD’, indica que el fichero ya debe existir en el ordenador. Si no lo encuentra dará un mensaje de error. ‘NEW’, indica que es un fichero nuevo que debe ser creado por el programa. Si existiera uno con ese nombre aparecería un mensaje de error. ‘UNKNOWN’ indica que el estatus es desconocido y que si existe lo toma como tal y si no lo crea.

Por ejemplo

```
OPEN (unit=4, File =’datos.dat’, status =’unknown’)
```

indica al ordenador que debe abrir un fichero al que se le asigna el número de unidad 4 que se llama datos.dat y que su estatus es desconocido, es decir que si no existe lo crea.

La lectura desde el fichero que hemos definido se llevará a cabo poniendo la unidad en el primer campo

```
READ(4,*)A,B
```

Lee los datos A,B desde el fichero identificado por la unidad 4 en la sentencia open, en este caso el fichero datos.dat.

Cuando leemos un vector o una matriz necesitamos combinar la senticia READ/WRITE con la acción de repetir indicada por la sentencia DO. Existen distintos tipos de combinaciones.

Así por ejemplo

```
DO I=1,n
READ(*,*)A(I)
END DO
```

lee en pantalla las componentes de un vector de N elementos.

Más explicativa es la combinación

```
DO I=1,N
WRITE(*,*) 'Escribe la componente', I
READ(*,*)A(I)
END DO
```

que nos dará un mensaje pidiendo la componente correspondiente.

La combinación

```
DO I=1,N
WRITE(*,*)A(I)
END DO
```

Escribe en columna los elementos del vector A

Si ponemos

```
DO I=1,N
WRITE(*,*)I,A(I)
END DO
```

Escribirá una tabla de dos columnas con el índice y el elemento del vector.

El DO y las sentencias READ/WRITE pueden combinarse de forma implícita de la manera siguiente

```
READ(*,*) (A(I),I=1,N)
```

Lee las componentes del vector A que deben estar escritas en una misma fila

```
WRITE(*,*) (A(I),I=1,N)
```

Escribe el vector fila A, suponiendo que el vector A(i) contenga el cuadrado de i aparecería

1. 4. 9. 16. 25.

Podemos realizar otras combinaciones, así por ejemplo

WRITE(*,*) (I,A(I),I=1,N)

Escribe el índice I y luego la componentes del vector A(I)

1 1. 2 4. 3 9. 4 16. 5 25.

Podemos hacerlo más explicativo insertando texto

WRITE(*,*) (‘ a(‘,I,’)=’,A(I),I=1,N)

Escribe el texto a(a continación el índice i a continuación el texto)= y a continuación el elemento a(i). En el ejemplo anterior obtendríamos

a(1)=1. a(2)= 4. a(3)=9. a (4)=16. a(5)=25.

4. Sobre como escribir/leer los datos

Un * en el segundo campo de una sentencia READ/WRITE indica que los datos serán leídos o escritos utilizando la opción por defecto es decir todas las cifras significativas que contenga la variable.

Esto puede alterarse utilizando un formato tanto de lectura como de escritura. La forma de hacerlo es asignarle una sentencia FORMAT a la sentencia READ/WRITE utilizando el segundo campo de la misma

WRITE(*,44) A,B

Indica escribe las variables A,B con el formato indicado en la sentencia 44.

La sentencia FORMAT consta de un número de sentencia que la relaciona con una o varias sentencias READ/WRITE y una serie de campos asociados a las variables a leer o escribir. En estos campos se indica el tipo de variable, el número total de caracteres que ocupa la misma y el número de cifras decimales.

Los indicadores más comunes son F para las variables reales, I para las enteras, E para las reales en notación científica, D para las doble precisión, y A para las alfanuméricas.

Van seguidas de 2 números separados por un punto. El primero indica el espacio total ocupado por la variable, contando el signo, el punto decimal y las especificidades de la notación científica (ver ejemplo) y el segundo el número de cifras decimales.

En el caso de los formatos I, A no aparece el segundo campo, y sólo se indica el espacio reservado para la variable correspondiente.

Aparte de estos indicadores se encuentran las opciones X para insertar espacios y ‘….’ para insertar texto. La opción X va precedida de un número que indica los espacios a insertar (5X inserta 5 espacios) la opción ‘ …’ utilizada en escritura simplemente escribe el texto insertado entre comillas.

Así por ejemplo

44 FORMAT(F8.3,2X,F6.2)

Indica que escribimos las variables A, B de la sentencia Write anterior la primera ocupando 8 espacios con tres decimales y la segunda ocupando 8 espacios con 2 decimales.

Suponiendo que los valores de esas variables fueran A=-1.414257 y B=3.141592 escribiría

bb-1.414**bb**bb3.14

donde b indica espacio en blanco. Nótese que los dos blancos simbolizados en negrita se incorporan como consecuencia del format 2X.

Si hubieramos puesto

44 FORMAT(E12.5,4X,E10.3)

Habríamos obtenido

-0.14143E+01**bbbb**b0.314E+01

El formato

44 FORMAT('primera=',F8.3,'segunda=', E10.3)

Hubiera ofrecido

primera= bb-1.414segunda=b0.314E+01

4. Sobre como leer de un fichero sin conocer su tamaño

En ocasiones resulta necesario leer de un fichero del que desconocemos su tamaño, por ejemplo una tabla en dos columnas de un conjunto de datos experimentales pero no sabemos cuantos hay y sería engorroso ponerse a contarlos.

Las sentencias READ/WRITE ofrecen más opciones habitualmente no necesarias para un inicio en la programación pero que pueden ser de interés en un estado más avanzado. Una de estas opciones es la de fin de fichero. La idea es la de mandar a leer del fichero de forma reiterada hasta que llegue al fin y que el ordenador continúe los cálculos sin dar un error. Ello lo hacemos añadiendo un tercer campo END= seguido de un número de sentencia lo cual indica que si el fichero se ha terminado la ejecución del programa continúa en la línea con que lleva ese número.

Así por ejemplo

READ(1,*,END=20)

Indica que si se termina el fichero la ejecución del programa continúa en la sentencia etiquetada como 20. Así por ejemplo tenemos un fichero de dos columnas datos.dat pero del que no sabemos cuantos elementos (sabemos que menos que 10000)

```
    OPEN(UNIT=1,FILE='DATOS.DAT',STATUS='OLD')
    K=0
    DO I=1,10000
    READ(1,*,END=15) X(I),Y(I)
    K=K+1
    END DO
15  WRITE(*,*) 'Terminada lectura de fichero, tiene', K, ' elementos'
```

Donde además hemos insertado un contador K para saber cuantos elementos tiene.