Capítulo 1

Formatos y sus usos

En el capítulo anterior vimos los parámetros de las sentencias que manejan el sistema de lectura y escritura de datos. Pero indicamos que existe una sentencia asociada a un READ() o WRITE() que permite regular la cantidad de dígitos y dígitos decimales que ingresan o egresan al programa y su ubicación dentro del renglón el cual se lee. En este capítulo veremos el uso de esta función, aunque limitada a los comandos mas usados[1].

1.1. Sentencia Format()

Esta sentencia se usa en combinación con un READ() o WRITE() de la siguiente manera.

```
      READ(*,100) X,Y,Z
 100  FORMAT(códigos numéricos y de posicionamiento)
      WRITE(22,203) A,B,C
 203  FORMAT(códigos numéricos y de posicionamiento)
```

Como pueden ver en el ejemplo, el format se escribe en una sentencia aparte. Esta sentencia está numerada y y puede estar en cualquier lugar del programa, pero no puede faltar si su número figura en un read() o en un write(). Tampoco significa que la ejecución del programa salta a ese lugar en particular. La sentencia es usada para indicar detalles de ese ingreso o egreso de datos. Varios read() o write() pueden usar la misma sentencia sentencia format. El número de etiqueta sirve para saber cuál es la sentencia format usada en ese lugar en particular.

1.2. Códigos para datos

Los códigos numéricos indican la cantidad de cifras que nos interesan de un número en particular. Es decir cuántas cifras ingresan al programa aunque, por ejemplo, que el número que se lee tenga más cifras que las que indica como necesarias. Es decir, se lee la indicación y los dígitos sobrantes se descartan. Según el tipo de variables tendremos que usar un formato diferente. Veamos los principales:

1.2.1. Variables enteras ->Formato I

Se utiliza de la formar In donde n es la cantidad de cifras decimales de interés. Por lo cual un formato sería escrito como I3,I5,I6, etc. Si leo varias variables enteras, lo debería hacer de la forma:

[1] Existen muchas variantes de estas órdenes, siempre se puede buscar en un manual de Fortran algún comando muy específico, no es de interés de la cátedra su conocimiento exhaustivo

Uso del código "I" para leer números

```
      READ(*,122) jj1,jj2,jj3,jim
 122  FORMAT(I3,I4,I2,i6)
```

Y el programa corre y cuando llega el read() se escribe:

```
1234567890123456789
```

Se leerá en la variable jj1 un número con formato I3, que significa que. tiene 3 dígitos entonces el número 123.
Para la variable siguiente la jj2, esta tiene indicado un formato I4, es decir se leerán los 4 dígitos que siguen, es decir el número 4567.
En el caso de la tercer variable, la jj3 esta tiene formato I2, o sea se lee los dos dígitos siguientes, por lo cual el número leído es: 89
Y en la variable final jim el formato es I6, entonces el número es 012345. De ahí en adelante los números no asignados a a variables se descartan.

Uso del código "I" para escribir números

Si el programa ejecuta la siguientes sentencias:

```
      Write(*,102) ki,kj,kz
  102 FORMAT(I3,I4,i6)
```

Y dentro de las variables en binario se encuentran guardados los números 145 en la variable ki, 456 en kj y 7890 en kz. Estos se escribirán en la pantalla como:

```
145␣789␣␣7890
```

Note que al ser el formato más grande que el número, el lugar se ocupa con espacios en blanco indicados como: ␣

Resumiendo: si el número es más chico que el tamaño del formato se imprimirá con espacios en blanco en su parte izquierda para completar la cantidad de dígitos indicados en el código I. Si el número tiene más dígitos que el formato indicado se señalará el error con una cantidad de "*" igual al formato de la sentencia.
En el ejemplo anterior si el formato hubiese sido:

```
      Write(*,102) ki, kj, kz
  102 FORMAT(I3,I2,i6)
```

El resultado al correr el programa es:

```
145 **  7890
```

Hay que tener en cuenta que así como un espacio en blanco ocupa un lugar, el signo del número también lo hace. Normalmente no se indica que el número es positivo, por ejemplo: +54, aunque para la computadora se lo puede hacer. Sólo estamos acostumbrados a indicar el signo cuando el número es negativo. Por ejemplo, si tengo el número -22 en la memoria de la computadora, deberé escribirlo con formato de al menos I3. Un dígito sería para el signo (-) y dos para el número 22.

Forma multiplicativa de los código de datos

Esta forma se usa para hacer la sentencia forma más compacta: si fácil de entender con un ejemplo:

```
      Write(*,102) ima, ima1,ima2, ima3, ima4
  102 FORMAT(I3,I4,I4,I4)
```

Que se puede escribir de la forma multiplicativa equivalente:

```
      Write(*,102) ima, ima1, ima2, ima3, ima4
  102 FORMAT(I3,3I4)
```

3I4 significa 3 veces I4.
Y que funciona incluso agrupando códigos:

```
  102 FORMAT(I3,I4,i2,I4,I2)

  102 FORMAT(I3,2(I4,I2))
```

2(I4,I2) significa 2 veces I4,I2.

1.2.2. Variables real sin exponente ->Formato F

Este formato se usa para los número reales (o flotantes) y lleva consigo dos valores. Se escribe como “Fn.i” , donde el “n” es igual que en los enteros, este número es la cantidad de cifras totales que tiene le número y el “i” es la cantidad cifras decimales. Por ejemplo, el número F7.3 indica que el número ocupa 7 lugares y 3 de estos son decimales (ojo, que aquí el punto cuenta). Un número F7.3 sería 123.456
Si leo un número con este formato y este número ya tiene un punto puesto, la parte de la indicación decimal se ignora y se ingresa el número tal como se lo leyó. Por ejemplo, si el formato es F7.3 y leo el número 1234.56, el número queda tal como está. Si el número fuese 1234.5678 este quedará leído como 1234.56 [2], ya que se cuentan 7 cifras contando el punto. En cambio, si el número no tiene el punto decimal el formato se lo agrega. Ejemplo: Si ingreso el número 123456, el número queda leído como 123.467
En caso de la escritura de un valor con este formato se escribirá con la la exacta cantidad de decimales, Si estos son más que la indicación se procederá al redondeado del número para que quede con los decimales solicitado.

1.2.3. Variables real con exponente ->Formato E

En su uso es muy parecido al formato “F"pero difiere en los números son leídos esperando que haya una indicación de notación científica es dicer, se esperan número como 1.234E-34. Este formato se escribo como “En.i” donde “n” e “i” funciona igual que en el formato F. En este formato ya que hay que considerar la notación científica se escribirá ocupando lugares y por lo tanto el valor “n” (que da la cantidad total de cifras del número) debe ser mucho más grande que “i”. Ejemplo: E12.4
Hay más formatos en FORTRAN pero los más comunes y usados son los que hemos visto. Se deja la curiosidad del lector el estudio de otros de estos códigos.

[2] Estrictamente muchos sistema modernos ingresaría el número 1234.57 (redondeado) que el 1234.56 (truncado) para minimizar el error en la magnitud del número. Ya que el 1234.57 es más cercano que el 1234.56 al valor ingresado 1234.5678

1.3. Códigos de Posicionamiento

Aparte de los códigos que indican cómo escribir un número existen códigos que son para indicar donde los escribo, por eso esos códigos se los clasifica como de posicionamiento, no indican ninguna característica del número sino su ubicación. En este apunte no daremos detalle de estos códigos salvo por el código “X”.
El código “X” indica un espacio en blanco que se deja al escribir o leer, por lo cual, por ejemplo, 3X indicaría 3 espacios en blanco.