# 2.1.- EJEMPLO DE UN PROGRAMA FORTRAN

Con el presente apartado comenzaremos a conocer cómo se escribe un programa en lenguaje FORTRAN bajo el entorno de programación FORTRAN.

En primer lugar conozcamos que aspecto presenta un programa sencillo que se encargará de calcular el área de un rectángulo para ir descubriendo las reglas básicas de escritura que debemos mantener. El programa ha sido escrito bajo un símil de papel cuadriculado, siendo equivalente cada cuadrícula a un carácter de escritura. Este aspecto no es el de la pantalla pues no aparece cuadriculada; sin embargo esta forma de mostrar el programa nos ayudará a comprender las reglas de escritura del código FORTRAN. El número de columnas que a continuación se presentan está limitado a 30 para evitar prolongar la hoja de escritura innecesariamente; sin embargo es preciso tener en cuenta que en FORTRAN está permitido escribir hasta la columna 72. Para conocer en todo momento en que línea y columna nos encontramos el entorno de programación ofrece en la ventana de edición en su parte inferior derecha dos números, uno correspondiente a la línea y otro a la columna de la posición del cursor por lo que se hace muy fácil conocer este aspecto.

| NUMERO DE COLUMNA DENTRO DE UNA LINEA DE PROGRAMA | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 13 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

```
      PROGRAM AREAS
C
C     ESTE PROGRAMA CALCULA EL
C     AREA DE UN RECTANGULO
C
      IMPLICIT NONE
      REAL a, b, area
      PRINT *, 'DAME EL LADO a'
      READ *, a
      PRINT *, 'DAME EL LADO b'
      READ *, b
      area = a * b
      PRINT *, 'AREA = ', area
      END
```

Aunque más adelante se irán describiendo los comandos y sentencias FORTRAN, para entender el programa anterior es preciso introducir el significado de, al menos, aquellos elementos del FORTRAN presentes en el programa; por ello se describen brevemente a continuación.

**PROGRAM :** Es la primera sentencia FORTRAN que aparece en un programa y siempre está acompañada de un nombre que denomina de forma simbólica el nombre del programa. Su inclusión es opcional; sin embargo un

código bien escrito debería incluirla para que cuando sea revisado en un futuro conozcamos su nombre.

A continuación en el programa aparecen tres líneas cuyo primer carácter es una C y que son comentarios dentro de las cuales se describe la función del programa AREA. No son pues instrucciones que el programa deba ejecutar. Esta sección también es opcional; sin embargo su inclusión es muy recomendable puesto que permite a cualquier programador conocer la función del código con el que está tratando.

**IMPLICIT NONE:** Sentencia FORTRAN que obliga al programador a declarar todas y cada una de las variables utilizadas en el código, de este modo ninguna variable será considerada declarada. implícitamente. Mas adelante será explicado este concepto extensamente.

**REAL:** Sentencia FORTRAN que permite asignar a las variables a, b, c su calidad de números reales. Esta sección es la correspondiente a la declaración de variables.

**PRINT:** Esta sentencia presenta su formato más elemental y se encarga de imprimir, en este caso en pantalla, el contenido entrecomillado que aparece a su derecha. Obsérvese que en la segunda utilización de este comando, además de imprimir lo entrecomillado, mostrará el valor de la variable area como resultado. Es importante poner especial atención a la utilización de comas como símbolo separador de variables fundamentalmente.

**READ:** Sentencia equivalente a PRINT, pero para leer datos. En este caso leerá las variables a y b que le sean facilitadas al programa por la pantalla.

**END:** Es la última sentencia FORTRAN y significa que el programa FORTRAN ha finalizado y que por lo tanto su ejecución debe terminar en ese punto.

## 2.2.- NORMAS DE ESCRITURA DE UN PROGRAMA FORTRAN

En el ejemplo anterior puede observarse un cierto orden de escritura de los comandos y sentencias de FORTRAN; esto se debe a que existen una serie de normas que se deben cumplir ya que de lo contrario el compilador no podrá interpretar correctamente el código escrito. Estas normas son sencillas y fundamentalmente hacen referencia a la colocación de los comandos dentro de la hoja de escritura FORTRAN. A continuación se describen las normas fundamentales que debemos cumplir.

***Línea de programa**:* Se denomina así a un conjunto de caracteres tomados del conjunto de instrucciones FORTRAN (comandos, funciones, variables, etc.). Cada carácter ocupa una columna y las columnas se enumeran de izquierda a derecha. Cada línea de programa únicamente podrá incluir una sentencia FORTRAN.
A continuación se muestra el entorno de edición cuadriculado por filas y columnas para una mejor comprensión de las particularidades de las columnas 1, 2, 3, 4, 5, 6 , 7 y sucesivas, durante la edición del código.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ..... columnas siguientes |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| :<br>:<br>filas siguientes | | | | | | | | | |

***Línea de comentario**:* Es aquella línea en la que el programador se permite incluir comentarios o mensajes propios. Estas líneas no son *líneas de programa*, y tiene la particularidad de que son siempre líneas que empiezan con un C en la columna 1 de su propia línea. De este modo el programa compilador detecta automáticamente que es una línea que no debe tener en cuenta y no la compila.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ..... columnas siguientes |
|---|---|---|---|---|---|---|---|---|---|
| C | Línea de comentario | | | | | | | | |
| : | | | | | | | | | |
| : | | | | | | | | | |
| filas siguientes | | | | | | | | | |

***Línea de número o etiqueta de una sentencia**:* Esta línea está formada por dos partes: el número entero de uno a cinco dígitos que deberá estar escrito dentro de las columnas 1 a 5 y la sentencia propia de FORTRAN que deberá estar escrita de la columna 7 en adelante.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ..... columnas siguientes |
|---|---|---|---|---|---|---|---|---|---|
| *Número o Etiqueta* | | | | | | *Sentencia FORTRAN* | | | |
| : | | | | | | | | | |
| : | | | | | | | | | |
| filas siguientes | | | | | | | | | |

***Línea de programa***: Esta línea incluye cualquiera de las sentencias FORTRAN carentes de etiqueta. Son las más comunes dentro de un programa y deberán aparecer escritas de la columna 7 en adelante.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ..... columnas siguientes |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | *Sentencia FORTRAN* | | | |
| : | | | | | | | | | |
| : | | | | | | | | | |
| filas siguientes | | | | | | | | | |

***Línea de continuación***: En ocasiones las sentencias FORTRAN pueden llegar a ser tan largas que el número de caracteres excede del permitido para una misma línea (72), por lo que nos vemos obligados a saltar a la siguiente línea para continuar con la sentencia. Esta circunstancia especial se resuelve incluyendo cualquier carácter en la columna 6 y continuando la sentencia fraccionada.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | .....columnas siguientes |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | *Sentencia* <========== | | | |
| | | | | | # | ==========> *FORTRAN* | | | |
| : | | | | | | | | | |
| : | | | | | | | | | |
| filas siguientes | | | | | | | | | |

## EJERCICIO 2.1

1. Abrir el entorno de programación FORTRAN, copiar el programa areas mostrado anteriormente, tal y como está escrito, teniendo mucho cuidado en los espacios, comas y demás caracteres.

2. Una vez copiado, guardarlo en el directorio C:\TEMPORAL con el nombre AREAS.FOR
3. Compilar el programa.
4. Construir el ejecutable.
5. Correr el programa y comprobar numéricamente la validez de los resultados.
6. Una vez comprobado el correcto funcionamiento, guardar el programa en A:\sesion2

# 2.3.- TIPOS DE DATOS: CONSTANTES Y VARIABLES

En primer lugar mostraremos los tipo de constantes con los que FORTRAN puede trabajar. En el apartado de variables se verán las variables en FORTRAN, o lo que es lo mismo que nombres y de que tipo deben ser las variables encargadas de tomar los valores de las constantes previamente definidas. En este apartado se deberá prestar atención especial a como se declaran este tipo de variables para el caso de que nuestro código precise emplearlas y así poder declararlas adecuadamente en la sección del programa dedicada a la declaración de variables. Podemos recordar que en el programa ´areas´ ya estudiado, el bloque en el que se declaraban las variables estaba formado por la sentencia REAL. En cualquier caso es importante tener presente de aquí en adelante que todas las variables han de ser declaradas una por una y que el bloque dedicado a la declaración de variables deberá ir precedido siempre de la sentencia IMPLICIT NONE.

**2.3.1.- CONSTANTES ENTERAS:** Una constante entera es una sucesión de dígitos precedidos o no del signo positivo (+) o negativo (-) y sin coma decimal. Si la constante es positiva, no es necesario escribir delante de ella el signo +. Por otro lado una constante entera puede ser nula, es decir puede tomar el valor 0.

| | | | | | | |
|---|---|---|---|---|---|---|
| ***Ejemplos:*** | 34 | -987 | +45 | 0 | 23456 | 00023 |

**2.3.2.- CONSTANTES REALES:** Este tipo de dato equivale a un numero real ordinario, es decir una parte entera más una parte fraccionaria separadas por un punto decimal y no por una coma. Este tipo de dato se subdivide en tres grupos:

**a) Constante real Básica**
**b) Constante real básica con exponente**
**c) Constante entera con exponente.**

Este tipo de constantes reales se denominaban definido

**a) Constante Real Básica:** Está formada por un signo (+ opcional) seguido de una serie de dígitos, un punto y una nueva serie de dígitos.

| | | | | | | |
|---|---|---|---|---|---|---|
| ***Ejemplos:*** | 34.098 | -0.0097 | 43.=(43,0) | .87=(0,87) | - .0023= (- 0,0023) | 456.0 |

**b) Constante Real Básica con Exponente:** Se define exponente real en FORTRAN, al carácter alfabético **E**

seguido de un signo (+ opcional) y por una constante entera formada por dos dígitos como máximo.

| | | | |
|---|---|---|---|
| ***Ejemplos:*** | 6.456E0 = 6,456 x $10^{0}$ | - 4.33E-06 = - 4,33 x $10^{-6}$ | 5.56E+3 = 5,56 x $10^{3}$ |

La parte que precede al carácter E se denomina mantisa, mientras que la constante entera que define el valor exponencial se denomina característica.

**c) Constante Entera con Exponente:** Se define como una constante entera seguida de un exponente real.

| | | | |
|---|---|---|---|
| ***Ejemplos:*** | + 456E-03= 456,0 x $10^{-3}$ | - 12E2 = - 12,0 x $10^{2}$ | 45E00 = 45,0 x $10^{0}$ = 45,0 |

Las constantes reales definidas hasta ahora se conocen con el nombre de constantes de punto (o coma) flotante ya que la posición del punto decimal depende de cual sea el valor del exponente o característica.

## 2.3.3.- CONSTANTES CARACTER:

Se denomina constante carácter a un conjunto de uno o más caracteres FORTRAN válidos y su longitud es el número total de caracteres que contiene. Una constante carácter debe encerrarse entre unos delimitadores; en este caso se emplea el apóstrofe (´) ( no confundir con las comillas).

| | | |
|---|---|---|
| ***Ejemplos:*** | ´ ESTO ES UNA CONSTANTE CARACTER ´ | ´ El VALOR DEL AREA ES =´ |

## 2.3.4.- CONSTANTES DOBLE PRECISION:

Se denomina exponente doble precisión en FORTRAN, al carácter alfabético D seguido opcionalmente de un signo + o – y finalizado por una constante entera formada por dos dígitos como máximo. Existen dos clases de constantes doble precisión que son: constantes reales básicas con exponente doble precisión y las constantes enteras con exponente doble precisión. Su similitud con las de simple precisión es decir, la previamente mostradas, es total y la única diferencia es que el compilador reserva más espacio de memoria para almacenar las de doble que las de simple precisión, lo que implica que estas últimas conservan menos cifras decimales significativas que las de doble precisión.

| | | | |
|---|---|---|---|
| ***Ejemplos:*** | 6.456D0 = 6,456 x $10^{0}$ | - 4.33D-06 = - 4,33 x $10^{-6}$ | - 12D2 = - 12,0 x $10^{2}$ |

## 2.3.5.- CONSTANTES COMPLEJAS:

Una constante compleja en FORTRAN es un par ordenado de constantes enteras o reales separadas por una coma y encerradas entre paréntesis. La primera parte representa la parte real mientras que la segunda representa la parte imaginaria.

| | | |
|---|---|---|
| ***Ejemplos:*** | ( 6.45, - 4.56 ) = 6.45 – 4.56 i | ( 6.5E1, 4.5 ) = 65 + 4.5 i |

## 2.3.6.- VARIABLES ENTERAS:

Las variables enteras pueden tomar cualquier nombre compuesto por uno o como máximo seis caracteres y designan zonas de memoria cuyo contenido es una constante entera. El primer carácter del nombre no podrá ser un número en ningún caso.

| | | | |
|---|---|---|---|
| ***Ejemplos:*** | AREA | TEMP1 | I |

Si las tres variables enteras anteriores se fueran a emplear en un programa su declaración vendría dada por:

| | |
|---|---|
| ***Declaración:*** | INTEGER AREA,TEMP1,I |

Si deseamos asignar una constante entera a una variable del mismo tipo:

TEMP = - 987

Como un vestigio de los primeros lenguajes FORTRAN existen compiladores que, en caso de no haber sido declaradas las variables pero si usadas durante el código, interpretan su tipo real o entero dependiendo del primer carácter. Esta particularidad por regla general, provoca errores difíciles de descubrir. Por ello, para evitarlo, se recomienda incluir en la línea de programa precedente a la declaración de variables la sentencia:

IMPLICIT NONE

Con ello se garantiza que el programa no asignará un tipo determinado a ninguna variable, obligándonos a declarar todas las variables adecuadamente, lo que es norma indispensable de un buen programador.

## 2.3.7.- VARIABLES REALES:

En cuanto a su concepción y nomenclatura son idénticas a las variables enteras, únicamente se diferencian en que designan un espacio en memoria cuyo contenido es una constante real. Así podríamos tener nuevamente los mismos ejemplos:

| | | | |
|---|---|---|---|
| ***Ejemplos:*** | AREA | TEMP1 | I |

Si ahora deseamos que las variables alberguen constante reales su declaración debería ser::

| | |
|---|---|
| ***Declaración:*** | REAL AREA,TEMP1,I |

Si deseamos asignar una constante real a una variable del mismo tipo:

TEMP = - 4.33E-06

## 2.3.8.- VARIABLES LOGICAS:

Son variables un tanto especiales y cuyo cometido es tomar únicamente dos valores cierto o falso (1 o 0), en cuanto

a su nomenclatura es igual a las anteriores.

| ***Ejemplos:*** | CORTE | ONOF | LOG |
|---|---|---|---|

Si ahora deseamos que las variables sena lógicas su declaración debería ser::

| ***Declaración:*** | LOGICAL CORTE,ONOF,LOG |
|---|---|

## 2.3.9.- VARIABLES CARACTER:

Como su nombre indica sirven para almacenar constantes tipo carácter y su nomenclatura vuelve a ser idéntica a las anteriores así podemos tener los siguientes ejemplos:

| ***Ejemplos:*** | PALAB | CADENA | X |
|---|---|---|---|

A diferencia con las anteriores declaraciones se ha de determinar en la sentencia de declaración la longitud de la constante carácter que va albergar: Esto puede verse a continuación:

| ***Declaración:*** | CHARACTER*10 PALAB |
|---|---|
| ***Declaración:*** | CHARACTER*3 CADENA,X |

Si deseamos asignar una constante carácter a una variable del mismo tipo:

FRASE = ´ESTO ES UNA CADENA´

Lo anterior significaría que la variable PALAB podrá almacenar cadenas de caracteres, es decir constantes carácter, de longitud máxima igual a 10, mientras que las variables CADENA y X solamente podrán almacenar constantes carácter de longitud 3. Si en la declaración no se especifica el número que indica la longitud permitida el compilador tomará por defecto longitud uno.

## 2.3.10.- VARIABLES DOBLE PRECISION:

Ya definimos anteriormente las constantes de doble precisión, y por lo tanto de forma equivalente deberán declararse aquellas variables que soporten constantes de ese tipo; así podemos ver los siguientes ejemplos:

| ***Ejemplos:*** | XYZ | RADIO | AS |
|---|---|---|---|
| ***Declaración:*** | DOUBLE PRECISION XYZ,RADIO,AS | | |

Si deseamos asignar una constante de doble precisión a una variable del mismo tipo:
RADIO = - 4.33D-06

### 2.3.11.- VARIABLES COMPLEJAS:

Su nomenclatura mantiene los criterios seguidos hasta ahora. Así podríamos tener los siguientes ejemplos:

| | | | |
|---|---|---|---|
| ***Ejemplos:*** | COPOS | VELOS | S |

En cuanto a su declaración, decir que la palabra COMPLEX define aquellas variables que durante el programa serán tratadas como complejas.

| | |
|---|---|
| ***Declaración:*** | COMPLEX COPOS, VELOS, S |

Si deseamos asignar una constante compleja a una variable del mismo tipo:

COPOS = CMPLX( 2.1,3.4) con lo que COPOS almacenara el complejo 2.1+3.4 i

## EJERCICIO 2.2

1. Modificar el programa AREA para que trabaje exclusivamente con enteros
2. Compilar, crear el ejecutable y ejecutarlo.
3. Intentar introducir valores por pantalla reales y ver lo que ocurre.
4. Intentar introducir un carácter como dato y ver lo que pasa.
5. Con tus conocimientos actuales ¿qué medidas podrían establecerse para que el usuario del programa no cometa errores al introducir los datos?.

## EJERCICIO 2.3

1. Desarrollar un programa que calcule el área de un círculo.

## EJERCICIO 2.4

1. Desarrollar un programa que, sin pedir ningún tipo de dato de entrada, genere y escriba en pantalla un número real, un entero, un complejo y un carácter. Estos datos deberán ser siempre los mismos y deberán estar declarados y definidos en el programa