UNIDAD X

FORTRAN APLICADO A LA RESOLUCION DE ECUACIONES ALGEBRAICAS Y TRASCENDENTES.

INTRODUCCION.

En las unidades anteriores estudiamos las reglas del lenguaje FØRTRAN y dimos ejemplos de su uso. Es evidente que una buena familiaridad con estas reglas, junto con algo de ingenio, nos permite escribir una variedad de programas para la solución de problemas más o menos complicados. Sin embargo, existen ciertos problemas que sólo se pueden resolver, por medio del computador, cuando se conocen ciertos métodos matemáticos. Permitámonos entonces colocar el arte de la programación en un medio más realista; supongamos que se nos presenta un problema de ingeniería el cual tenemos que resolver, pero que no conocemos el método matemático que se necesita para su solución. Primero debemos aprender el método matemático y luego debemos deducir cómo lo programamos. Finalmente, debemos determinar si es necesario introducir algunas modificaciones para ajustar el programa a los requisitos del problema.

Uno de los problemas que se encuentran frecuentemente en ingeniería es la determinación de raices de una ecuación de la forma

$$f(x) = 0$$

En esta unidad se estudiarán algunos métodos para determinar las raices reales de dichas ecuaciones.

OBJETIVOS.

Al terminar esta unidad deberás ser capaz de:

1. Describir perfectamente en que consiste cada uno de los métodos estudiados para resolver ecuaciones de la forma $f(x) = 0$.

2. Resolver problemas en los cuales se tenga que encontrar las raices de una ecuación de la forma $f(x) = 0$, en forma manual y por computadora.

3. Comparar la eficiencia computacional de los métodos para resolver ecuaciones de la forma $f(x) = 0$.

PROCEDIMIENTO DE APRENDIZAJE.

1. Estudiar el anexo de la unidad.

2. Consultar el libro: JAMES, SMITH, WOLFORD, METODOS NUMERICOS APLICADOS A LA COMPUTACION con FORTRAN. Pág. 141 – 152.

Requisito:

Para presentar el examen de esta unidad deberás entregar un problema resuelto por computadora en donde uses alguno de los métodos aprendidos, de preferencia, de la autoevaluación o de los ejercicios.

EXAMEN DE AUTOEVALUACION.

Podrás pedir examen de evaluación cuando puedas resolver el siguiente cuestionario.

1. Dada la ecuación

$$y = x^3 + 3.5x^2 + 7.5x + 3.0$$

   a. Decir cuantas raices tiene.

   b. Usando calculadora de escritorio hacer los primeros 5 pasos para determinar alguna de sus raices empleando el método de tanteos.

   c. Hacer el diagrama de flujo y programa para la computadora digital para determinar las raices de la ecuación por el método de tanteos.

2. Por el método de Newton-Raphson determina la raíz de

$$x^3 - 0.39x^2 - 10.5x + 11.0$$

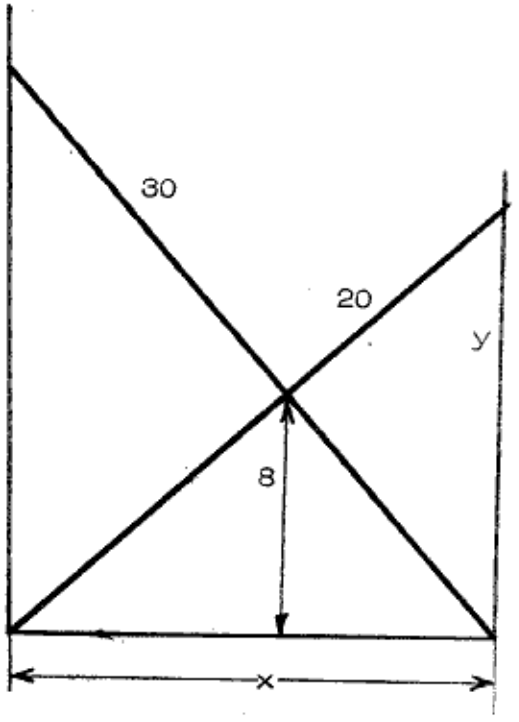
   comprendida entre 2 y 3.

3. Dos escaleras, una de 20 pies de longitud y la otra de 30 pies están apoyadas contra las paredes de un callejón, como se muestra en el diagrama. Si el punto en el que se cruzan las escaleras está situado a una altura de 8 pies, ¿cuál es el ancho del callejón?.

En su libro "problems for computer solution" (publicsdo en 1964), Gruenberger y Jaffrey demuestran que la formulación de este problema puede requerir la solución de la siguiente ecuación:

$$y^4 - 16y^3 + 500y^2 - 8000y + 3200 = 0$$

entonces $x = \sqrt{400 - y^2}$



UNIDAD X

MATERIAL ANEXO.

Uno de los problemas de ocurrencia más frecuente en ingeniería es hallar las raíces de ecuaciones de la forma

$$f(x) = 0 \qquad \ldots\ldots \quad (1)$$

Donde la función $f(x)$ puede estar dada explícitamente, como, por ejemplo, un polinomio en X o como una función trascendente. Con frecuencia, sin embargo, $f(x)$ puede conocerse solo implícitamente; puede ser conocida una regla para calcular $f(x)$ para cualquier argumento, pero su forma explícita es desconocida. En casos raros puede ser posible obtener las raíces exactas, por ejemplo, en un polinomio factorizable. En general, sin embargo, podemos esperar la obtención de soluciones aproximadas solamente contando con alguna técnica computacional para producir la aproximación. Cualquiera que sea el método computacional a usar, primero se hace un análisis matemático para conocer el dominio de la función y el rango donde se encuentra la raíz.

Método de Tanteos :

En el método de tanteos determinamos valores de $f(x)$ correspondientes a valores sucesivos de X hasta que presenta un cambio de signo en $f(x)$. El cambio de signo indica que se ha pasado por una raíz. Entonces se puede obtener una mayor aproximación al valor de la raíz volviendo al último valor de X que precede el cambio de signo y, a partir de este valor de X, determinar nuevamente valores de $f(x)$ correspondientes a valores sucesivos de X, utilizando ahora un incremento menor que el que se usó inicialmente, hasta que cambie de nuevo el signo de $f(x)$. Este procedimiento se repite con incrementos de X cada vez más pequeños hasta lograr un valor suficientemente preciso de la raíz. Si se desean raices adicionales, se repite todo el algoritmo desde dar una nueva X (o sea el nuevo intervalo encontrado en el análisis matemático).

Se debe tener cuidado en :

1. Dar valores iniciales muy grandes a X, para que se pueda pasar una raíz en el caso de que tenga dos raices muy próximas, se deben utilizar incrementos pequeños.

2. Que la computadora se detenga a causa de sobre flujo, si el valor de la función tiende a volverse infinitamente grande para determinados valores de X. No utilizar valores tan pequeños.

Ejemplo: Calcular la primera raíz que satisfaga la ecuación

$$f(x) = x^3 - x - 1 = 0$$

1. Hacer un análisis de la curva para valores de:

X	f(x)
= 0	−1
< 0	< 0
> 0	−1 ....
X = 1	−1
X = 2	5
X > 2	> 0

para este ejemplo se halla que

$$f(1) = -1 < 0 < 5 = f(2)$$

por lo tanto, como f(x) es continua, f(x) debe anularse en alguna parte en el intervalo $[1, 2]$. Entonces, como $f'(x) = 3x^2 - 1$ es positiva sobre $[1, 2]$, f(x) tiene exactamente una raíz en el intervalo $[1, 2]$.

2. En seguida se procede a determinar valores de f(x). Para valores de X comprendidos entre 1 y 2. En este caso se considera un valor del incremento de X (ΔX) = 0.1. De acuerdo a lo anterior se construye la siguiente tabla.

X	f(x)
1	−1
1.1	−0.769
1.2	−0.472
1.3	−0.103
1.4	+0.344

Se observa un cambio de signo entre −0.103 , 0.344 . Continuando el proceso, determinamos valores de f(x) para valores de X comprendidos entre el intervalo. Dividimos $\Delta X/10$ y obtenemos el nuevo intervalo

$$\Delta X = 0.01$$

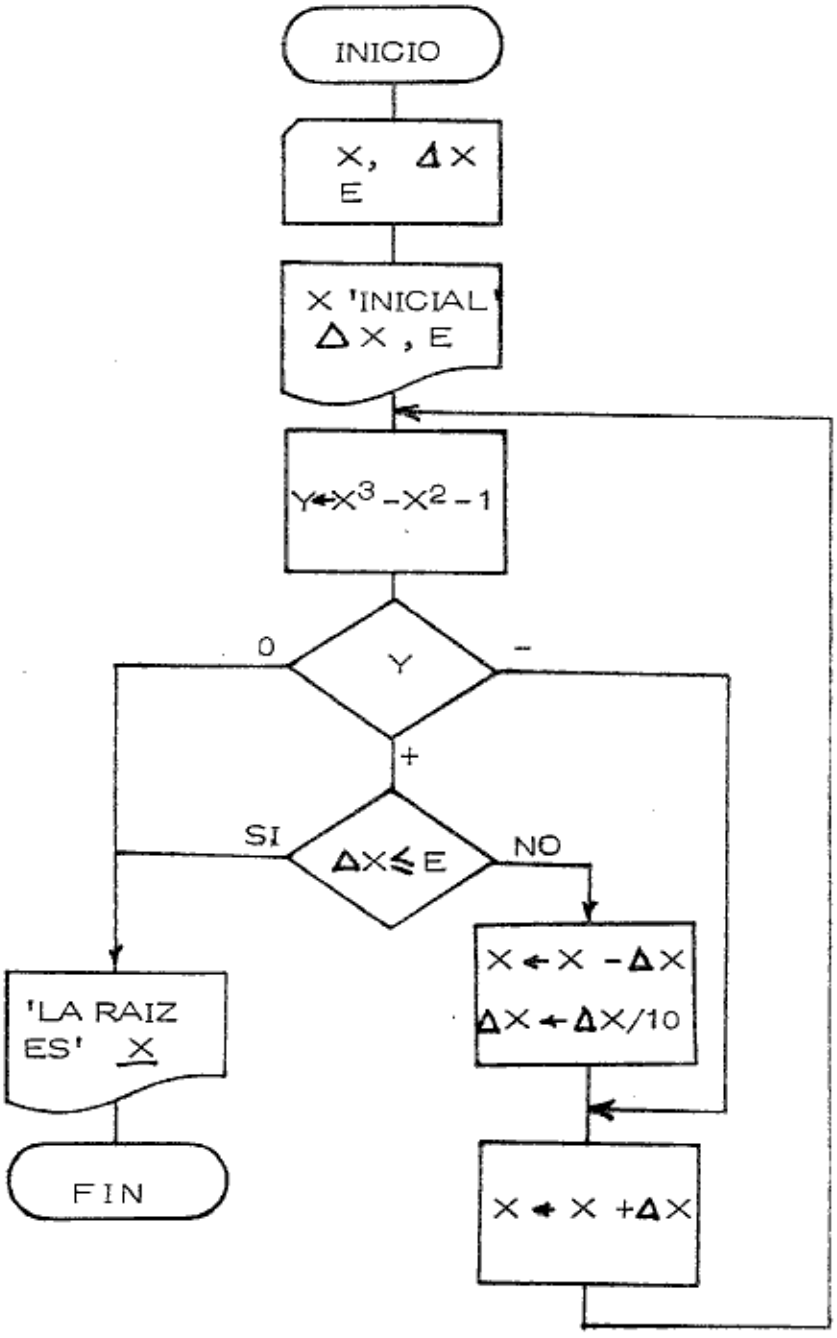
Los valores obtenidos son:

X	f(x)
1.3	−0.103
1.31	−0.062
1.32	−0.020
1.33	0.0226

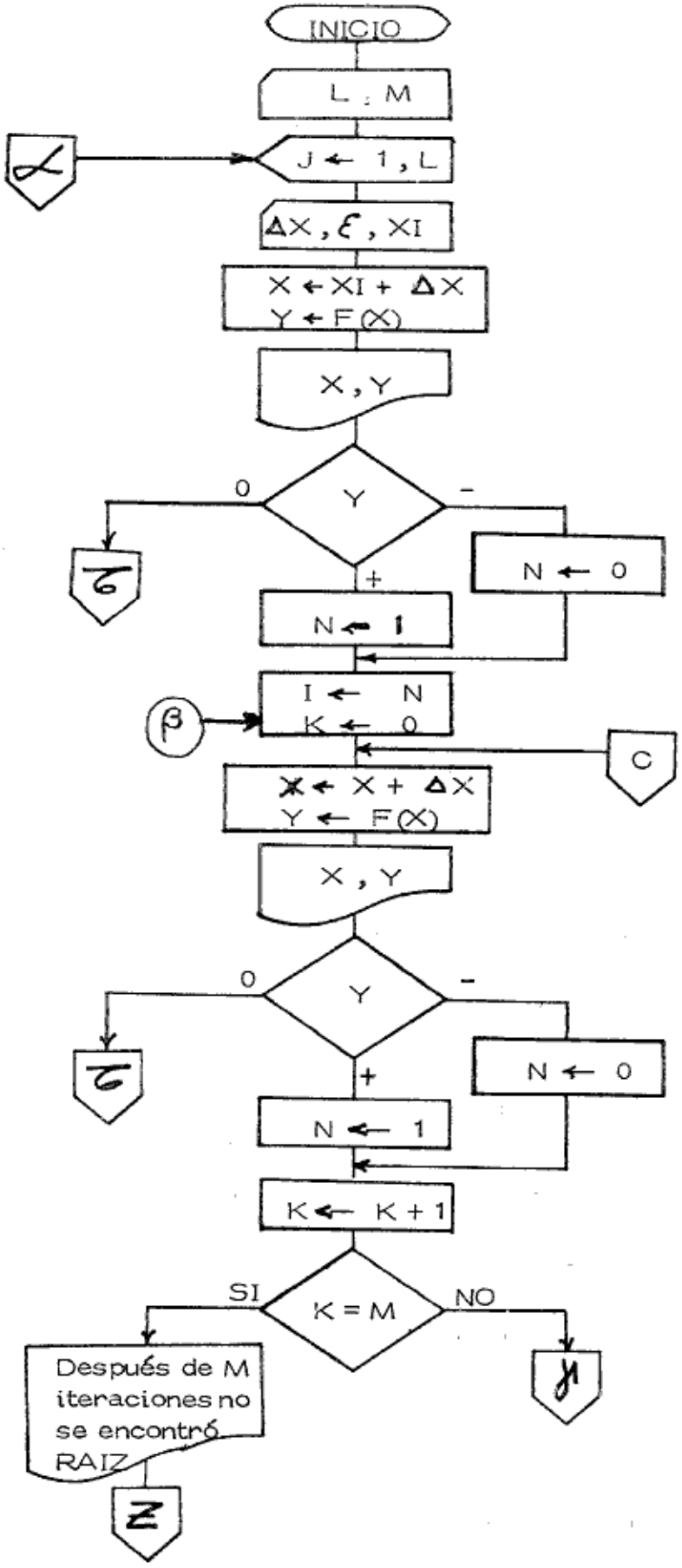
y se observa que el valor de la raíz está comprendido entre -0.020 y 0.0226.

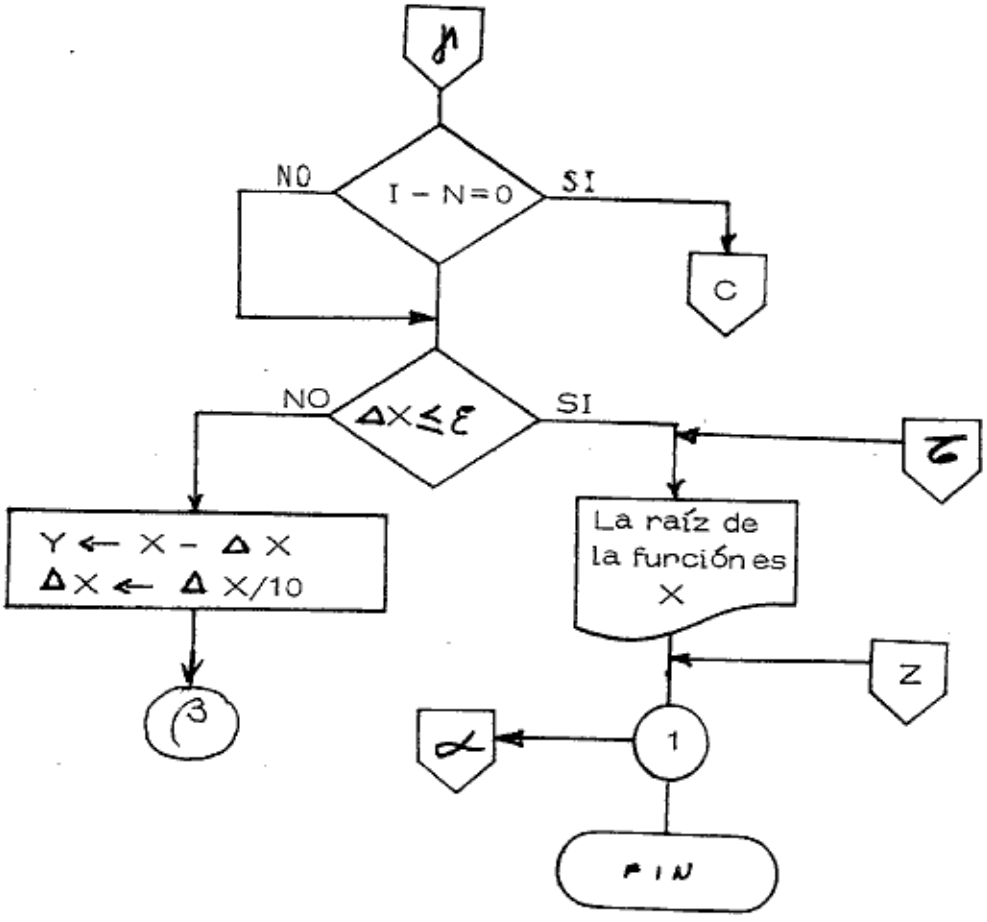
El proceso se continua hasta obtener la precisión deseada.

El diagrama de flujo para este problema sería:



Un diagrama de flujo que sirve para calcular raices de funciones ó de polinomios es :




β
I - N=0
NO
SI
C
ΔX≤ε
NO
SI
2
Y ← X - Δ X
Δ X ← Δ X/10
La raíz de la función es X
Z
3
1
α
FIN

Codifica el diagrama de flujo. En cada paso importante, escribe los comentarios que sean necesarios para que quede una codificación muy completa.

Notas y Comentarios.

L = grado del polinomio o número de raices que se quieren calcular.

M = número de iteraciones deseadas para calcular c/u de las raices.

ΔX = incremento de X en el intervalo.

$\mathcal{E}$ = precisión deseada de la raíz.

N e I = nos permiten saber si la función cambió de signo.

K = contador para el número de iteraciones deseadas.

J = número de raices encontradas.

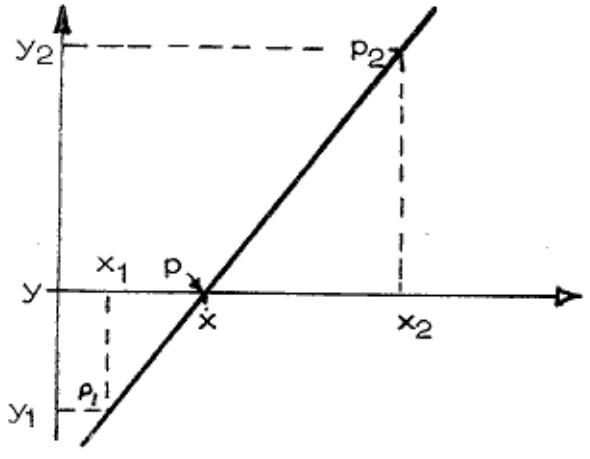
XI = X inicial del intervalo

F(X) = valor de la función ya sea trascendente ó un polinomio.

Y = valor calculado de la función.

Método de las Posiciones Falsas:

Este método determina las raices de una ecuación, con un grado especificado de precisión, y con un número menor de operaciones de computadora que las que se utilizan en el método de tanteos. Las dos primeras aproximaciones al valor de la raíz se obtienen por el método de tanteos, trás de lo cual se utiliza una interpolación lineal entre dos puntos, tales como p_1 y p_2



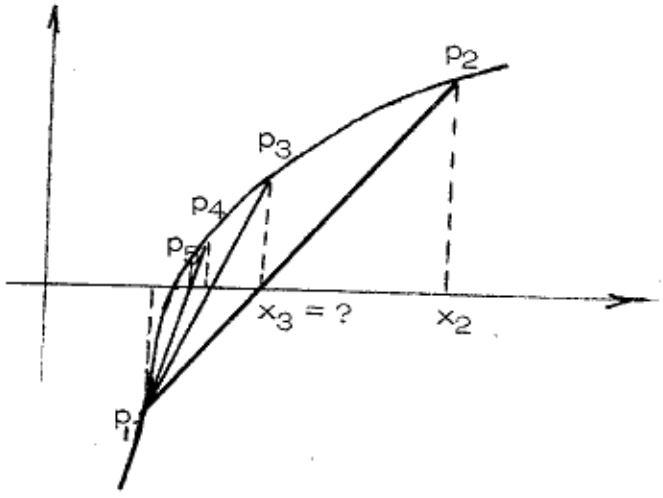
Analizando la figura se tiene:

$$\frac{y - y_1}{x - x_1} = m = \frac{y_2 - y_1}{x_2 - x_1}$$

Si en el punto P, se considera que $y = 0$:

$$X = \frac{x_1 y_2 - x_2 y_1}{y_2 - y_1} \qquad \cdots\cdots \qquad (2)$$

Aplicado a una curva se tiene:



El valor de X se puede determinar a partir de la ecuación 2, ya que se conocen todos los valores del segundo miembro. El valor que se determina para X_3 se sustituye entonces en la función, para determinar el valor de y_3. A esto sigue una interpolación entre los puntos P_1 y P_3, de la cual se obtiene el valor de X_4, que es aún una mejor aproximación. El proceso se repite son pares sucesivos de puntos hasta que se llega a una aproximación que tenga la precisión deseada. Cuando dos valores consecutivos de X difieren en una cantidad menor que cierto valor de prueba de precisión ξ, se ha logrado la precisión deseada.

Método de Newton-Raphson:

Este método es útil para mejorar una primera aproximación a una raíz de una ecuación de la forma $f(x) = 0$, que pudo haber sido obtenida por tanteos, por una gráfica aproximada a la función, o por algún otro método.

```
$CONTROL USLINIT
C PROGRAMA PARA APROXIMAR UNA FUNCION POR EL METODO DE INTERPOLACION
C                          LINEAL
C     ESTE METODO SIRVE PARA APROXIMAR UNA FUNCION A SUS RAICES POR EL
C     METODO DE INTERPOLACION LINEAL (REGULA-FALSI)
C     SIMBOLOGIA:
C L= GRADO DEL POLINOMIO
C M = NO. DE INTERPOLACIONES REQUERIDAS
C AX = INCREMENTO DE X EN EL INTERVALO
C                      (NO PONER = CERO)
C F = PRECISION QUE SE REQUIERA
C                IDEAL 0.0001
C X1= VALOR INICIAL DEL INTERVALO
C     LA FORMA DE DAR LOS DATOS ES:
C PRIMERA TARJETA
C     GDO. DEL POLINOMIO(I), INTERACIONES(K)
C     SEGUNDA TARJETA Y SIGUIENTES HASTA QUE EXISTA UNA PARA CADA
C     RAIZ (GDO. DEL POLINOMIO)
C     VALOR DE (XI),PRECISION(E),INCREMENTO(AX)
      READ(5,*)L,M
      DO 1 J=1,L
      READ(5,*)AX,F,XI
      WRITE(6,10)J,XI,AX,E
      X1=0.
      Y1=0.
      X2=0.
      Y2=0.
      X=XI
      K=0
   25 Y=2*X**2+1.-EXP(X)
      WRITE(6,20)X,Y
      IF(Y)21,22,23
   21 X2=X
      Y2=Y
      IF(X1.EQ.0..AND.Y1.EQ.0.) GO TO 24
   26 X3=(X1*Y2-X2*Y1)/(Y2-Y1)
      IF(X-X3.LE.E)GO TO 22
      X=X3
      GO TO 25
   23 X1=X
      Y1=Y
      IF(X2.NE.0..AND.Y2.NE.0.) GO TO 26
   24 K=K+1
      IF(K.EQ.M) GO TO 27
      X=X+AX
      GO TO 25
   27 WRITE(6,30)M
      GO TO 1
   22 WRITE(6,40)X
    1 CONTINUE
   10 FORMAT(/,20X,'LOS VALORES INICIALES PARA LA',I2,'RAIZ SON','XI=',
     *F6.2,'AX=',F6.2,'E=',F7.4,//)
   20 FORMAT(/,20X,'X=',F6.3,10X,'Y=',F6.3)
   30 FORMAT(/,20X,'DESPUES DE',I3,'ITERACIONES NO SE ENCONTRO RAIZ EN E
     *STE INTERVALO')
   40 FORMAT(/,20X,'LA RAIZ ENCONTRADA DE LA FUNCION ES',F7.4)
      STOP
      END
```

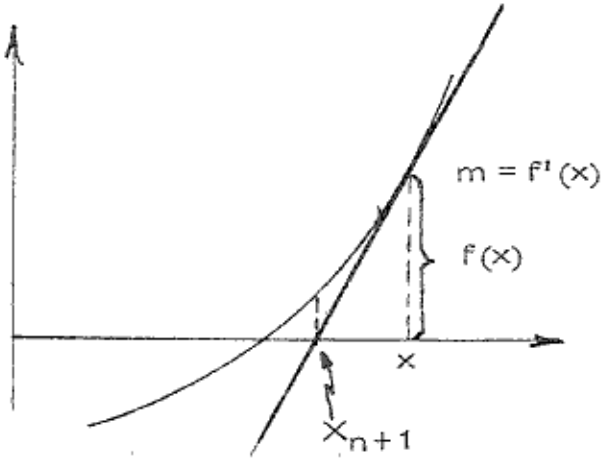
```
LOS VALORES INICIALES PARA LA 1RAIZ SONXI=  -.50AX=   .10E=  .0001

X= -.500          Y=  .893
X= -.400          Y=  .650
X= -.300          Y=  .439
X= -.200          Y=  .261
X= -.100          Y=  .115
X= -.000          Y=  .000
X=  .100          Y= -.085
X= -.000          Y=  .000
LA RAIZ ENCONTRADA DE LA FUNCION ES -.0000
LOS VALORES INICIALES PARA LA 2RAIZ SONXI=   .50AX=   .10E=  .0001

X=  .500          Y= -.149
X=  .600          Y= -.102
X=  .700          Y= -.034
X=  .800          Y=  .054
X=  .738          Y= -.002
LA RAIZ ENCONTRADA DE LA FUNCION ES  .7383
LOS VALORES INICIALES PARA LA 3RAIZ SONXI= 2.50AX=   .10E=  .0001

X= 2.500          Y= 1.318
X= 2.600          Y= 1.056
X= 2.700          Y=  .700
X= 2.800          Y=  .235
X= 2.900          Y= -.354
X= 2.840          Y=  .016
LA RAIZ ENCONTRADA DE LA FUNCION ES 2.8399
```

Considérese la gráfica de $f(x)$ en función de X;



y supongase una primera aproximación a una raíz. Si dibujamos una recta tangente a la curva en $X = X_n$, intersectará al eje X en un valor X_{n+1}, que constituye una aproximación mejorada a la raíz. Se puede observar que la pendiente de la tangente es

$$f'(X_n) = \frac{f(X_n)}{X_n - X_{n+1}} \qquad \ldots (3)$$

por lo tanto

$$X_{n+1} = X_n - \frac{f(X_n)}{f'(X_n)} \qquad \ldots (4)$$

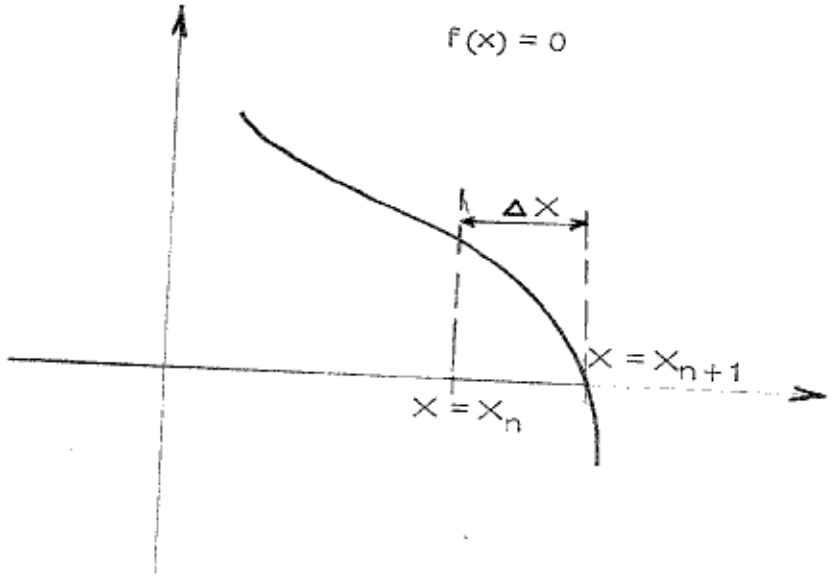
El valor de la función y el valor de la derivada de la función se conocen para $X = X_n$, y la nueva aproximación a la raíz, X_{n+1}, se obtiene utilizando la ecuación 4. Se repite el procedimiento con esta nueva aproximación, para obtener una mejor aproximación a la raíz. Esto continúa hasta que dos valores consecutivos de la raíz aproximada difieran en una cantidad menor que cierto valor epsilon prescrito, que controla el error permisible a la raíz.

Método de Segundo Orden de Newton:

Cuando se necesita determinar con mucha precisión el valor de una raíz de alguna ecuación, el método de segundo orden de Newton tiene la ventaja de rápida convergencia a una solución, que permite obtener una aproximación extremada--

mente cercana al valor de la raíz con un mínimo de cálculos. Sin embargo desde el punto de vista práctico, este método está limitado a su utilización en ecuaciones que tienen derivadas de mayor orden (cuando menos de 2º orden) relativamente simples, ya que el tiempo que se consume en obtener y programar derivadas complicadas sobrepasa la ventaja de la convergencia rápida.

Considérese una vez más una ecuación de la forma



Una gráfica de la función dibujada contra X se muestra en la figura. Supóngase que se ha determinado un valor aproximado de la raíz, $X = X_n$ por algún método tal como aproximación gráfica. Desarrollando f(x) en una serie de Taylor con respecto a $X = X_n$ se obtiene

$$f(X_{n+1}) = f(X_n) + f'(X_n)(\Delta X) + \frac{f''(X_n)(\Delta X)^2}{2!} + \frac{f'''(X_n)(\Delta X)^3}{3!} + \cdots \quad \cdots(5)$$

Si ΔX fuera el incremento particular de X que al sumarse a X_n, produjera un valor O para la serie, la cantidad $(X_n + \Delta X)$ sería exacta. Como nos interesa un procedimiento práctico para determinar un valor de ΔX que haga que la suma de la serie sea O, hagamos igual a O el segundo miembro de la ec. (5), utilizando únicamente tres términos de la serie. Obtenemos la igualdad aproximada.

$$f(X_n) + \Delta X \left[f'(X_n) + \frac{f''(X_n)\ \Delta X}{2} \right] = 0 \qquad \cdots\cdot (6)$$

Un valor de ΔX determinado a partir de la ec. (6) cuando se suma a X_n, no proporcionará el valor exacto de la raíz, ya que se utilizarán únicamente tres

términos de una serie infinita para valuar ΔX. Sin embargo, se tendrá una aproximación mucho mejor al valor de la raíz. De la ec. (3) se tenía

$$\Delta X = X_{n+1} - X_n$$

$$\Delta X = - \frac{f(X_n)}{f'(X_n)}$$

Sust. este valor en la ecuación (6)

$$f(X_n) + \Delta X \left[f'(X_n) - \frac{f''(X_n)\ f(X_n)}{2\, f'(X_n)} \right] = 0$$

despejando ΔX obtenemos :

$$\Delta X = - \frac{f(X_n)}{f'(X_n) - \left(\frac{f''(X_n)\ f(X_n)}{2\, f'(X_n)} \right)}$$

Como $\Delta X = X_{n+1} - X_n$

$$X_{n+1} = X_n - \left[\frac{f(X_n)}{f'(X_n) - \left(\frac{f''(X_n)\ f(X_n)}{2\, f'(X_n)} \right)} \right] \quad \ldots (7)$$

que se puede utilizar para aproximaciones cada vez más cercanos a una raíz.

Realiza el programa para resolver por computadora estos problemas.

Problema Nº 1.

Dada la ecuación

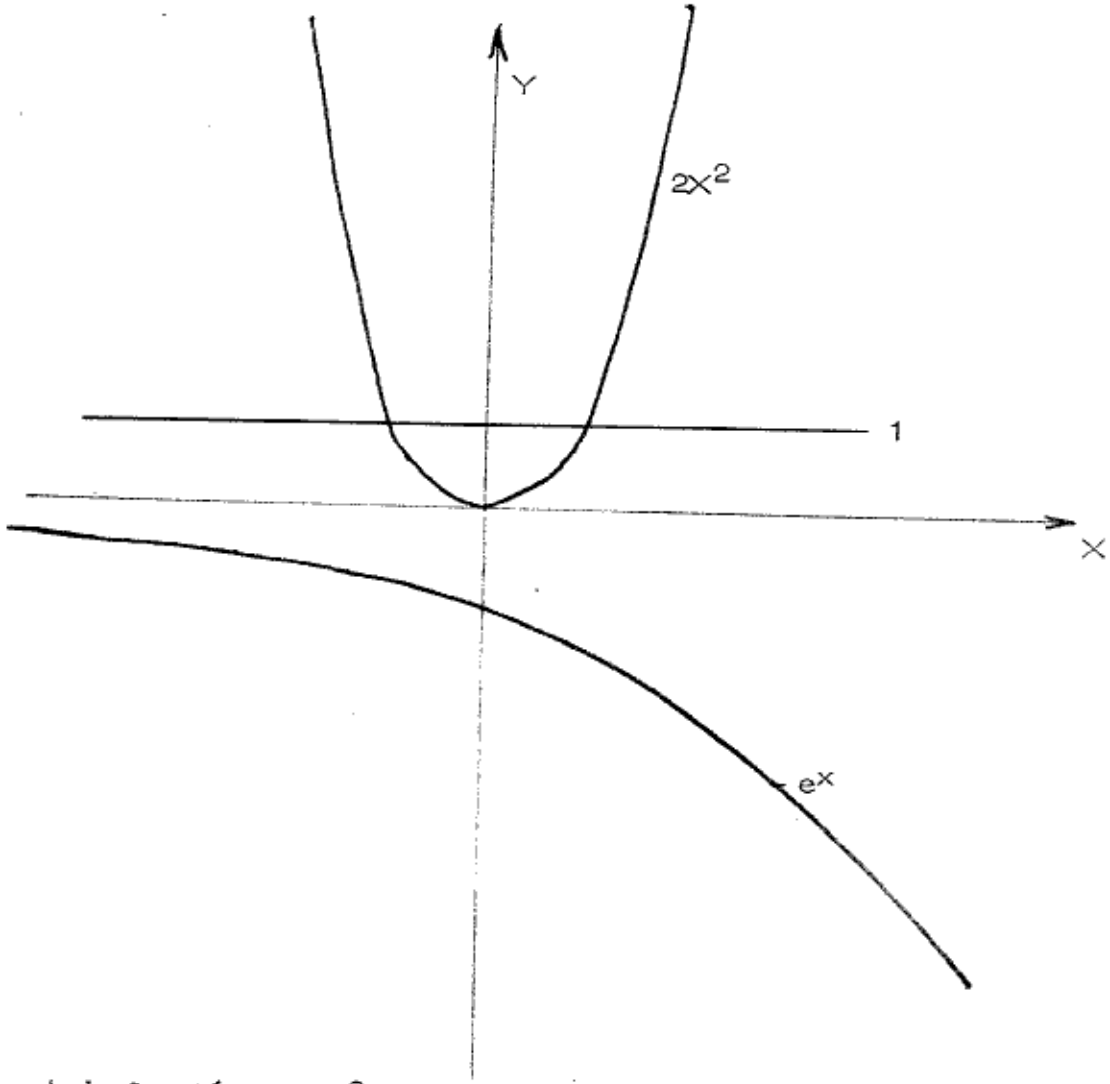
$$Y = 2X^2 + 1 - e^X$$

Hacer el análisis de la curva y determine la primera raíz positiva empleando el método de Newton-Raphson.

Análisis de la curva.

La función está definida en todo el eje real, es decir en el intervalo $(-\infty, \infty)$.

La gráfica de la función es como sigue:



El valor de la función $y = 2x^2 + 1 - e^x$ en cada $\underline{\underline{X}}$ es la suma de los valores de cada una de las funciones mostradas en la figura.

De acuerdo con lo anterior, si $x = 0$, $y = 0$ ó sea que $x = 0$ es una raíz.

Para valores negativos de $\underline{\underline{X}}$, entonces el valor de $\underline{\underline{Y}}$ siempre será positivo ya que:

$$2x^2 + 1 > 1 \quad \text{si} \quad x < 0$$

$$-e^{-x} < 1 \quad \text{si} \quad x < 0$$

y la suma será mayor que cero si $x < 0$.

Para valores positivos de X, entonces :

$2x^2 + 1 > 0$ y aumenta a medida que $\underline{X}$ aumenta

y

$e^{-x} < 0$ y disminuye a medida que $\underline{X}$ aumenta

De acuerdo a lo anterior es posible la existencia de raices positivas.

Con el objeto de tener una idea más clara del comportamiento de la función en el intervalo $(0, \infty)$.

Se determina el valor de la función en varios puntos.

X	$2X^2$	1	$-e^X$	Y	
0	0	1	-1	0	RAIZ
0.5	0.5	1	-1.648	-0.148	< 0
1.0	2.0	1	-2.7182	0.1818	> 0

O sea que entre 0.5 y 1.0 hay una raíz.

Como para valores muy grandes de $\underline{X}$

$$1 + 2X^2 < e^{-X}$$

se espera que haya otro camio de signo de Y.

Por lo tanto continuamos con nuestra tabla

X	$2X^2$	1	$-e^X$	Y
2.0	8.0	1.0	-7.389	$0.611 > 0$
3.0	1.8	1.0	-20.08	$-1.08 < 0$

o sea que entre X = 2 y X = 3 hay otra raíz.

Las ecuaciones empleadas para determinar las raices de una ecuación empleando el método de Newton-Raphson son :

Dada $Y = f(x)$ y $y' = f'(x)$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

y el proceso se interrumpe hasta que $\left| x_{n+1} - x_n \right| \leq \mathcal{E}$

Para nuestro ejemplo

$$y = 2x^2 + 1 - e^x$$
$$y' = 4x - e^x$$
$$\mathcal{E} = 0.0001$$

Para determinar la raíz que se encuentra entre $x = 0.5$ y $x = 1.0$ podemos tomar $x_o = \underline{0.5}$

$$x_o = 0.5$$

$$y(x_o) = 2 \times 0.5^2 + 1.0 - e^{0.5} = -0.1487$$

$$y'(x_o) = 4 \times 0.5 - e^{-0.5} = 0.3513$$

$$X_1 = 0.5 - \frac{-0.1487}{0.3513} = 0.9234$$

$$\left| X_1 - X_o \right| = \left| 0.9234 - 0.5 \right| = 0.4234$$

1er. Paso.

$$X_1 = 0.9234$$

$$y(X_1) = 2 \times 0.9234^2 + 1.0 - e^{0.9234} = 0.1875$$

$$y'(X_1) = 4 \times 0.9234 - e^{0.9234} = 1.1757$$

$$X_2 = 0.9234 - \frac{0.1875}{1.1757} = 0.7639$$

$$\left| X_2 - X_1 \right| = \left| 0.7639 - 0.9234 \right| = 0.1594$$

2º Paso.

$$X_2 = 0.7639$$

$$y(X_2) = 2 \times 0.7639^2 + 1 - e^{0.7639} = 0.0205$$

$$y'(X_2) = 4.0 \times 0.7639 - e^{0.7639} = 0.9090$$

$$X_3 = 0.7639 - \frac{0.0205}{0.9090} = 0.7414$$

$$\left| X_3 - X_2 \right| = \left| 0.7414 - 0.7639 \right| = 0.0225$$

3er. Paso.

$$X_3 = 0.7414$$

$$y(X_3) = 2 \times 0.7414^2 + 1 - e^{0.7414} = 0.0005$$

$$y'(X_3) = 4 \times 0.7414 - e^{0.7414} = 0.8667$$

$$X_4 = 0.7414 - \frac{0.0005}{0.8667} = 0.7409$$

$$\left| X_3 - X_4 \right| = \left| 0.7414 - 0.7409 \right| = 0.0005$$

$$X_4 = 0.7409$$

$$y(X_4) = 0.0$$

Por lo tanto la primera raíz positiva es :

$$X = 0.7409$$

Problema Nº 2.

Dada la ecuación del problema 1 determinar la primera raíz empleando el método de segundo orden de Newton.

El análisis de la curva se hizo en el problema número 1 .

Las ecuaciones del método de Newton de segundo orden son :

$$X_{n+1} = X_n - \left[\frac{f(X_n)}{f'(X_n) - \frac{f''(X_n)\, f(X_n)}{2 f'(X_n)}} \right]$$

$$\left| X_{n+1} - X_n \right| \leq \varepsilon$$

El proceso se continua hasta que se cumpla la segunda ecuación:

En nuestro caso tenemos

$$X_0 = 0.5$$

$$y = f(X_n) = 2X^2 + 1 - e^X$$

$$y' = f'(X_n) = 4X - e^X$$

$$y'' = f''(X_n) = 4 - e^X$$

1er. Paso.

$$X_0 = 0.5$$

$$f(X_0) = 2 \times 0.5^2 + 1 - e^{0.5} = -0.1487$$

$$f'(X_0) = 4 \times 0.5 - e^{0.5} = 0.3513$$

$$f''(X_0) = 4 - e^{0.5} = 2.3513$$

$$X_1 = X_0 - \left[\frac{-0.1487}{0.3513 - \frac{2.3513 \times (-0.1487)}{2 \times 0.3513}} \right] = 0.6752$$

$$\left| X_1 - X_0 \right| = \left| 0.6752 - 0.5 \right| = 0.1752$$

2º Paso.

$$X_1 = 0.6752$$

$$f(X_1) = 2 \times 0.6752^2 + 1 - e^{0.6752} = -0.0527$$

$$f'(X_1) = 4 \times 0.6752 - e^{0.6752} = 0.7363$$

$$f''(X_1) = 4 - e^{0.6752} = 2.0352$$

$$X_2 = X_1 - \left[\frac{-0.0527}{0.7363 - \frac{2.0356 \times (-0.0527)}{2 \times 0.7363}} \right] = 0.7403$$

$$\left| X_2 - X_1 \right| = \left| 0.7403 - 0.6752 \right| = 0.0651$$

3er. Paso.

$$X_2 = 0.7403$$

$$f(X_2) = 2 \times 0.7403 + 1 - e^{0.7403} = -0.0005$$

$$f'(X_2) = 4 \times 0.7403 - e^{0.7403} = 0.8645$$

$$f''(X_2) = 4 - e^{0.7403} = 1.9035$$

$$X_3 = X_2 - \left[\frac{-0.0005}{0.8645 - \frac{1.9035 \times (-0.0005)}{2 \times 0.8645}} \right] = 0.7409$$

$$\left| X_3 - X_2 \right| = \left| 0.7409 - 0.7403 \right| = 0.0005$$

4º Paso.

$$X_3 = 0.7409$$

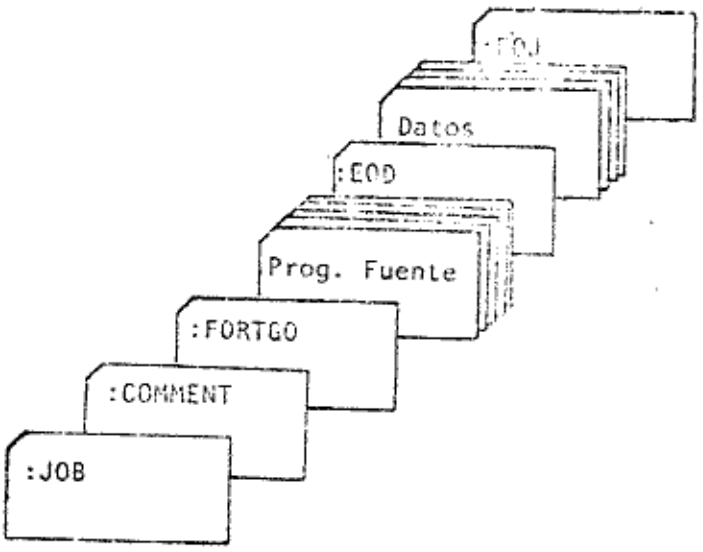
$$f(X_3) = 2 \times 0.7409^2 + 1 - e^{0.7409} = 0$$

Así la raíz positiva es

$$X = 0.7409$$

BOLETIN INFORMATIVO

TARJETAS DE CONTROL PARA USUARIOS DE FORTRAN



FORMATO DE TARJETA DE JOB.

:JOB Nombre del programa,USUARIO/Password de usuario.CUENTA/Password de cuenta,TIME=40

EJEMPLO:

Nombre del programa=TAREA1

Usuario=S4123456

Password de usuario=ALUMN56

Cuenta=A1121501

Password de cuenta=CONJUNTO

```
:JOB TAREA1,S4123456/ALUMN56.A1121501/CONJUNTO;TIME=40
:COMMENT   ERNESTO MARTINEZ NORIEGA
:FORTGO
$CONTROL USLINIT
      READ (5,200) A,B,C
      D=SQRT (A**B+C)
      WRITE (6,300) D
 200  FORMAT (F12.3)
 300  FORMAT ("RESULTADO:",F12.3/)
      STOP
      END
:EOD
12345.6
345.67
5.678
:EOJ
```

NOTA: Para cualquier consulta, Biblioteca cuenta con 3 juegos completos de manuales de HP-3000.

MENSAJES DE ERROR DEL COMPILADOR FORTRAN HP-3000

La primera parte de esta lista contiene los errores de sintaxis detectados por el compilador; para cada error se indica el número, el texto del mensaje en inglés, su traducción al español, y en ocasiones una descripción breve del motivo del error. Los errores de ejecución están listados en la última página.

0.- COMPILATION TERMINATED -Compilación suspendida -Ha ocurrido algún error que - hace que se suspenda la compilación. El compilador agrega a este mensaje la - causa de la suspensión. Corrija todos los errores y vuelva a meter su programa.

1.- NON-DIGIT IN LABEL FIELD -Caracter raro o alfabético en el campo de etiqueta - -En el campo de etiqueta solo puede haber números. El compilador ignora este - campo. Puede deberse a una proposición perforada a la izquierda de la columna 7.

3.- SYMBOLIC NAME EXCEEDS 15 CHARACTERS -Un nombre de variable se excede de 15 caracteres. Un nombre de variable tiene mas de 15 caracteres. El compilador trunca el nombre a 15. Cheque todos los nombres y asegurese que esten bien.

4.- EXPECTED A COMMA -Esperaba una "coma" -El compilador detecta que falta una coma, Si marca "Warning" la compilación continúa como si la "coma" estuviera incluida. Si marca error, requiere la coma.

5.- EXTRANEOUS COMMA - ', ' extraña -El compilador detecta coma inecesaria y la ignora. Corrija el programa para futuras compilaciones.

7.- UNEXPECTED CHARACTER -Caracter inesperado -El compilador detecta un caracter - inesperado y lo ignora. Examine la proposición y quítele el caracter extra.

8.- UNEXPECTED '-' —— '-' inesperado -El compilador detecta un guión o signo menos en un FORMAT, quitelo de la declaración en caso necesario.

9.- UNEXPECTED COMMA -'coma' inesperada -El compilador detecta una coma inesperada en un FORMAT. quitela de la declaración si es necesario.

10.- UNEXPECTED ')' —— ')' inesperado -El compilador detecta un parentesis inesperado en un FORMAT. quitelo de la declaración si es necesario.

11.- EXPECTED AN INTEGER -esperaba un entero -El compilador esperaba encontrar un entero en la declaración FORMAT. Sin cambiar la declaración la corrige, si es - causado por un Warning. Si es error, requiere el entero. Cheque la declara- - ción; si es necesario, inceste apropiadamente el entero y vuelva a meter su programa.

12.- EXPECTED A '.' —— '.' esperado -Si es Warning, el compilador esperaba un punto decimal en el FORMAT: Sin cambiar la declaración la corrige. Si es error en - un símbolo como .OR. requiere corregirlo y volver a meter el programa.

13.- EXPECTED A 'P' —— 'P' esperada -El compilador esperaba un factor de escala en el FORMAT. Sin cambiar la declaración la corrigue. Cheque la declaración si es necesario.

14.- UNEXPECTED 'P' —— 'P' inesperada -El compilador encontró un factor de identificación de escala que no esperaba en el FORMAT. Sin cambiar la declaración la corrige, cheque la declaración si es necesario.

15.- NESTING EXCEEDS 5 LEVELS -Se excede de 5 niveles anidados -Hay unas de 5 niveles de parentesis anidados en un FORMAT. Revise la declaración de tal manera que no queden mas de 5 niveles.

16.- EXTRANEOUS SOURCE -Extraña fuente- el compilador encontro un símbolo que no corresponde lógicamente a la proposición. El compilador borra el símbolo - y el texto que sigue. Examine la declaración y haga los cambios necesarios.

18.- SYMBOLIC NAME REDUNDANTLY TYPED -Nombre simbólico tipeado redundantemente. - A un nombre simbólico se le ha asignado el mismo tipo más de una vez. El - compilador no toma acción, quite la declaración redundante.

19.- SYMBOLIC NAME REDUNDANTLY EXTERNALLED -Nombre simbólico redundante en un - external -un nombre simbólico aparece en más de una declaración EXTERNAL. El compilador no toma acción. quite la declaración external extra.

20.- EXTRANEOUS EQUATE GROUP - Grupo de equivalencias extraño -una lista de parametros de un EQUIVALENCE contiene sólo un paramétro. incerte más variables - con el mismo almacenamiento.

22.- EXTRA INITIAL VALUES - Valores iniciales extras -DATA contiene más valores - que variables. Cheque la declaración y quite las variables o datos extra.

23.- EXTRANEOUS DATA ITEM -Dato extraño- el compilador descubrió un nombre de - variable en el DATA en un subprograma BLOCK DATA que no está tambien en un - COMMON. Cheque el DATA y el COMMON por nombres de variables.

24.- NO INITIAL VALUES -No hay valores iniciales - el compilador descubrió un sub programa BLOCK DATA que no especificaba valores iniciales para una declaración COMMON. Cheque ambas declaraciones por nombres de variables.

25.- INITIAL VALUE TRUNCATED -Valor inicial truncado - el compilador truncó un valor inicial porque fue un valor grande para este tipo. Determine el valor - correcto y remplácelo en el DATA.

26.- STATEMENT FUNCTION DUMMY USED AS NON -SIMPLE VARIABLE. -Parámetro en una declaración de función se utiliza como variable no sencilla -un parámetro en una declaración de función es referida en otra parte del subprograma no como una simple variable, por ejemplo como arreglo o un procedimiento.*

27.- EXPRESSION VS. NON -EXPRESSION ARGUMENT -Expresiones contra no- expresiones argumento- El compilador descubrió inconsistencia entre la estrucutra del argumento de la llamada de la subrutina y la subrutina de este programa.*

28.- SUBPROGRAMA VS. NON-SUBPROGRAM ARGUMENT- Subprogramas contra argumentos del subprograma -El compilador descubrió inconsistencia entre éste y la llamada de subprogramas previos en este programa, semejante al nombre del subprograma pasado.*

29.- SUBROUTINE VS. FUNCTION ARGUMENT -Subrutina contra argumentos de función- El compilador descubrió inconsistencia entre los argumentos en dos llamadas de subprograma en este programa.*

30.- ARGUMENT TYPE INCONSISTENT - Tipo de argumento inconsistente- llamadas hechas en la misma subrutina referida al mismo argumento mudo con diferentes tipos de argumento.*

* Revise el subprograma y la llamada de él, cambie lo que sea necesario y -- vuelva a meter su programa.

31.- SUBPROGRAM NAME NOT EXTERNALLED - Nombre del subprograma no externado- el - compilador descubrió un nombre de un subprograma no mencionado en un EXTERNAL.

32.- ARGUMENTS OF NESTED REFERENCE NOT CHECKED -Los argumentos en la referencia - no concuerdan - el compilador descubrió una llamada recursiva en la lista de argumentos actuales. Llama e informa usando sólo el argumento inicial.

33.- INTRINSIC NAME CONVERTED TO SIMPLE VARIABLE -Nombre intrinseco convertido a variable simple - el compilador convierte un nombre intrinseco a una variable simple, el nombre no puede referise a un subprograma remamente intrinseco.

34.- STATEMENT CANNOT BE REACHED -La declaración no puede ser alargada- el compilador descubrió una declaración que no puede ser alargada (por ejemplo una - declaración acompañan a un RETURN). Cheque la lógica del programa y añada - el número de declaración si es necesario.

36.- RETURN OR STOP INSERTED- El compilador inserta el postulado RETURN ó el STOP inmediatamente antes del ——. postulado END. Agregue el postulado faltante; de lo contrario, seguirá apareciendo este mensaje en cada compilación.

37.- BLANK LINE ILLEGAL- Existe una tarjeta la cual no tiene nada perforado, o -- sea, está en blanco; y para este compilador toda las tarjetas deberán tener una instrucción de lo contrario será ignorada. Para evitar el error hay - que eliminar dicha tarjeta: si realmente se olvidó poner la instrucción, añadala. Si la tarjeta en blanco tuviera una ubicación posterior a la del postulado END el compilador toma aquella como la proposición inicial de un subprograma en dado caso de que este existiera.

38.- TOO MANY CONTINUATION LINES -Demasiadas lines de continuación-Un postulado - puede contener hasta 20 lineas (la inicial y la de continuación). Si ocurre este error el compilador ignora totalmente el postulado.

39.- EXPECTED CONTINUATION LINE -El compilador esperaba una linea de continuación y esta no se encuentra.

41.- EXPECTED SYMBOLIC NAME -Se esperaba el nombre de una variable lo cual había sido previamente utilizada y no se encuentra. El compilador ignora este postulado.

43.- IMPROPER STATEMENT LABEL -El compilador descubre en la zona de etiquetas que esta mal perforada dicha etiqueta. Posiblemente tenga un simbolo diferente de un numérico.

49.- TITLE TOO LONG - Titulo demasiado grande. El titulo especificado en la tarjeta que contiene el comando $TITLE excede 52 caracteres; el compilador lo trunca a dicha cantidad.

50.- PROGRAM UNIT ABORTED -Programa abortado - El compilador termina con el programa que estaba siendo compilado, causado por un error y continúa con los siguientes programas.

51.- EXPECTED A '(' - El compilador esperaba un parentesis que abre ignorando con esto al postulado.

52.- EXPECTED A ')' -El compilador esperaba un parentesis que cierra y como no - sucede asi ignora el postulado.

53.- EXTRANEOUS ') ' -Existe un paréntesis que cierra de más. Dicho paréntesis es ignorado.

55.- EXPECTED A '] ' -El compilador esperaba un paréntesis cuadrado que cierra. Postulado ignorado.

56.- EXPECTED A ' [' -El compilador esperaba un parentesis cuadrado que abre. - Postulado ignorado.

58.- EXPECTED ASSIGNAMENT OPERATOR -El compilador esperaba un signo =. Esto no - ha ocurrido asi por lo que el postulado es ignorado.

59.- EXPECTED A ' / ' -El compilador esperaba una diagonal (/) por lo que ignora el postulado.

60.- EXPECTED A QUOTE -El compilador esperaba una tira de caracteres delimitada por apostrofes (') o por comillas ("). Tambien aquí el postulado es ignorado.

61.- EXPECTED A '\' -El compilador esperaba una diagonal inversa (\) en una - lista de parametros. Instrucción ignorada.

63.- IMPOSSIBLE CONTEXT FOR ' . ' -El compilador encontró un punto el cual no - encaja logicamente en la estructura sintáctica de la instrucción, por lo - cual, ésta es ignorada.

64.- IMPOSSIBLE CONTEXT FOR ' % ' -El compilador encontró un signo de porciento (%), el cual no encaja logicamente en la estructura sintáctica de la - instrucción, ignorandola el compilador.

65.- IMPOSSIBLE CONTEXT FOR ' $ ' -El compilador localizó un signo de dolares - ($) fuera de lugar, el cual es ignorado.

69.- STRING LITERAL EXCEEDS 255 CHARACTER -Una tira de caracteres entre apostrofes (') ó comillas (") tiene más de 255 caracteres.

76.- INTEGER CANNOT BE 0 -El compilador encontró un valor entero el cual no puede ser CERO (0).

77.- INTEGER EXCEEDS CONTEXTUAL LIMITS -El valor absoluto de un entero es demasiado grande en el contexto de la instrucción; el compilador, debido a esto, le asigna el valor de 1.

78.- INTEGER OVERFLOW -El valor de un entero revasa el limite de 32767. Utilice INTEGER*4 que permite un limite máximo de 214 748 364 7.

79.- EXPECTED EXPONENT VALUE -Exponente (E) ausente ó inválido.

80.- FLOATING LITERAL UNDER/OVERFLOW -Un valor de punto flotante (real) tiene un valor que cae fuera del rango 0.863617×15^{77} a 1.157921×10^{77}

84.- MISSING END LINE -Faltó el postulado END y el compilador lo pone por usted.

85.- UNEXPECTED CONTINUATION LINE. (inesperada continuación de linea). El compilador no espera continuación de la línea iniciada y elimina toda la proposición.

87.- REVERVED TOKEN NOT RECOGNIZED.-El compilador esperaba un símbolo reservado - tal como .OR ó .TRUE. y no encontró ninguno.

88.- CANNOT RECOGNIZE KEYWORD. (No puede reconocer palabra clave) -El compilador de Fortran/3000 espera una palabra clave tal como INTERGER OR REAL OR END y la proposición se suprimió al no ser reconocida la palabra clave. Modificar codificación y recompilar.

89.- CANNOT CLASSIFY STATEMENT. (Proposición no puede clasificarse) Posiblemente la proposición no comienza con una letra o se omitió en paréntesis derecho. - El compilador es incapaz de reconocer la proposición.

91.- STATEMENT OUT OF POSITION (Proposición fuera de posición) El compilador encontró una proposición de declaración que sigue a una proposición ejecutable; ó declaración fuera de orden en relación con otra proposición de declaración proposición suprimida.

92.- DUMMY NAME NOT UNIQUE. (Nombre mudo no único). -El compilador encuentra un - nombre de un parámetro mudo no único en el subprograma donde aparece.

93.- IMPROPER DUMMY ARGUMENT (Argumento mudo impropio) -El compilador encuentra - argumentos impropios y suspende verificación de lista de argumentos.

94.- ARGUMENT ADDRESSIBILITY EXCEEDED. (Direccionabilidad de argumentos excedida). -El compilador encuentra mas de 54 argumentos en la subrutina o función o - también muchos argumentos en postulado función (en postulado Fuction, numero de argumentos permitido depende del tipo y la complejidad de la expresión).

95.- TOO MANY ALTERNATE RETURNS (Demasiadas RETURNS). -El compilador encuentra - demasiados puntos de retorno en un subprograma.

96.- IMPROPER TYPE CONSTRUCT. (Construcción de tipo impropia). -El compilador ha encontrado una construcción de tipo incorrecta y suprimió la proposición.

97.- IMPROPER INITIAL LETTER CONSTRUCT. (Construcción de Letra inicial impropia). -Formato de proposición es Suprimida.

99.- SUBROUTINE CANNOT BE TYPED. (SUBROUTINE No puede tener declaración de tipo). -El compilador ha encontrado subprogramas de subrutina con una declaración de tipo explícito.

100.- SYMBOLIC NAME TYPED INCONSISTENTLY. (Nombre o Símbolo con tipo inconsistente) -El compilador ha encontrado un nombre ó símbolo cuyo tipo fue declarado de una manera inconsistente con una declaración previa.

102.- DYNAMIC BOUND DIMENSIONED. (Dimensión dinámica) -El compilador ha encontrado un arreglo dimensionado con una variable. Si se trata de un subprograma, la variable debe ser pasada en la lista de parámetros.

103.- PROCEDURE DIMENSIONED. (Procedimiento Dimensionado). -El compilador encontró el nombre de un subprograma en una proposición dimensión.

104.- ARRAY REDUNDATLY DEMENSIONED. (Arreglo redundantemente dimensiondado). -El - compilador encontró un arreglo cuya dimensión fue definida más de una vez - en el mismo programa.

105.- EXPECTED BOUND. (Esperaba dimensión de arreglo). -El compilador esperaba una una dimensión de un arreglo. Se suprime la proposición.

106.- ARRAY EXCEEDS 32767 ELEMENTS. (Arreglo excedido de 32767 elementos).-El compi lador encuentra un arreglo definido con más de 32767 elementos.

107.- NUMBER OF BOUNDS EXCEEDS 255. (Número de dimensiones excedido de 255). -El - compilador encuentra un arreglo con más de 255 dimensiones.

108.- DYNAMIC STRUCTURE IN COMMON. (Estructura dinámica en COMMON). -El compilador encuentra un arreglo de dimensión variable definido en una proposición COMMON.

109.- DUMMY NAME IN COMMON. (Nombre mudo en COMMON). El compilador encuentra nombres de parámetros mudos en una proposición COMMON.

110.- ITEM IN COMMON TWICE. (Variable en Common dos veces). -El compilador encuentra información que aparece en COMMON dos veces.

111.- COMMON BLOCK NAME ALSO PROCEDURF NAME. (NOMBRE COMMON BLOCK TAMBIEN NOMBRE DE PROCEDURE). -El compilador encuentra un nombre de bloque COMMON que es nombre de un subprograma.

112.- PROCEDURE NAME IN COMMON. (Nombre de procedure en COMMON). -El compilador encuentra un nombre de subprograma que aparece en COMMON.

113.- CHARACTER FUNCTION HAS DYNAMIC LENGTHE. (Función CHARACTER que tiene longitud Dinámica). El compilador encuentra una función CHARACTER definida como de - longitud dinámica (usando una especificadora de longitud variable).

114.- SIMPLE VARIABLE OR ARRAY EXTERNALLED. (Variable simple o arrego externo) -El compilador encuentra una variable simple o nombre de un arreglo usando en una proposición EQUIVALENCE.

117.- DUMMY NAME IN EQUATE. (Nombre mudo en EQUATE). -El compilador encuentra nombres de parámetros mudos en la proposición EQUIVALENCE.

118.- PROCEDURE NAME IN EQUATE. (Nombre de procedimiento en EQUATE). -El compilador encuentra un nombre de subprograma en una proposición EQUIVALENCE.

120.- DYNAMIC STRUCTURE IN DATA (Estructura dinámica en DATA). -El compilador encuentra una estructura dinámica en la proposición DATA.

121.- DUMMY NAME IN DATA (Nombre mudo en DATA). -El compilador encuentra el nombre de una parámetro mudo en la proposición DATA.

122.- PROCEDURE NAME IN DATA. (Nombre de procedure en Data). -El compilador encontró el nombre de un subprograma en proposición DATA.

123.- COMMON ITEM IN DATA ALLOWED ONLY IN BLOCK DATA. (Elemento de COMMON en DATA - aceptado solo en BLOCK DATA. -El elemento no debe aparecer en la proposición DATA.

124.- EXPECTED INITIAL VALUE. (Espera valor inicial) -El compilador espera un valor inicial para una variable. La proposición se suprime al no ser encontrado.

125.- INITIAL VALUE TYPE IMPROPER. (Valor inicial impropio). -El compilador encuentra un valor cuyo tipo no coincide con el tipo de la variable.

126.- UNARY SIGN REQUIRES ARITHMETIC LITERAL. (Signo unario requiere expresiones aritméticas). -El compilador encuentra un signo menos o un más no seguido por una constante.

127.- NUMBER OF SUBCRIPTS < > NUMBER OF BOUNDS. (Número de subscriptos < > número límites). -El compilador encuentra un elemento del arreglo con diferente número de subíndices que los que fueron especificados para el arreglo en la declaración Dimensión.

128.- ARRAY EXCEEDS 32767 ORDS. (Arreglo excedido de 32767 palabras). -El compilador encuentra que el arreglo ocupa más de 32767 palabras de memoria.

129.- SUBCRIPT VALUE NOT IN ARRAY. (Valor del subíndice fuera del arreglo). -El compilador encuentra un índice de un elemento del arreglo fuera de la definición de los límites del arreglo.

131.- LOCAL ADDRESSIBILITY EXCEEDED. (Dirección local, excedida). -El compilador fue incapaz de direccionar todas los variables locales en el programa.

132.- DYNAMIC BOUND NOT DUMMY INTEGER. (Límite Dinámico no entero mudo). -El compilador encuentra límite dinámico de un arreglo no representado por parámetro mudo entero).

134.- DATA BLOCK TOO LARGE. (Data Block muy grande). -El compilador encuentra que el programa requiere más de 32767 palabras para guardar todos los datos.

135.- COMMON BLOCK TOO LARGE.(Common Block muy grande). -El compilador encuentra COMMON BLOCK más grande que 32767 palabras.

136.- COMMON EXTENDED FORWARD.(Extensión COMMON hacia adelante). -El compilador encuentra una proposición equivalente que ha intentado extender el espacio de Common data.

137.- EQUATE BLOCK TOO LARGE. (Equate Block muy grande). -El compilador encuentra un grupo de proporciones EQUIVALENCE los cuales equivalen a un gran bloque de datos (mas que 32767 palabras).

141.- DATA BLOCK ITEM EQUATED TO COMMON BLOCK ITEM. -Variabe en DATA equivalente a variable COMMON - Una misma variable no puede aparecer al mismo tiempo en proposiciones COMMON y DATA; esto solo es valido en el subprograma BLOCK DATA.

144.- SIMPLE VARIABLE HAS SUBSCRIPT -Una variable sencilla tiene subíndice- Una variable que no es vector ni matriz ni arreglo de ningún tipo no puede lleva un subíndice.

145.- EXPECTED STATEMENT LABEL -Se esperaba un postulado con número (etiqueta)- El compilador esperaba un postulado con etiqueta y no lo encontró.

147.- DUPLICATE LABEL. -Etiqueta duplicada -El compilador se encontró dos postulados con el mismo número (etiqueta). El compilador ignora la segunda ocurrencia.

148.- UNRESOLVED LABEL REFERENCE. -Referencia a una etiqueta no resulta- El compilador se encontró una referencia a una etiqueta que no se le puso a ningún postulado.

149.- FORMAT REFERENCE TON NOM-FORMAT. -Referencia a un postulado que debería ser un formato y no lo es.

150.- EXECUTALBE REFERENCE TO NON-EXECUTALBE STATEMENT. -Referencia a un postulado que debería ser ejecutable pero no lo es.

153.- SUBROUTINE USED AS PRIMARY. -Subrutina usada como elemento primario- Una subrutina fué usada como una variable o un número en vez de ser llamada con un - postulado CALL.

154.- EXPECTED ARITHMETIC PRIMARY. -Se esperaba un elemento aritmético primario. -El compilador esperaba un elemento aritmético (número, variable, expr, etc.) y no lo encontró.

155.- NON-ARITHMETIC PRIMARY WHERE ARITHMETIC EXPECTED.-Elemento primario no aritmetico encontrado en donde se esperaba uno de tipo aritmetico.

156.- NON-LOGICAL OPERAND WHERE LOGICAL EXPECTED. -El compilador esperaba un operador lógico y se encontró uno que no lo es.

157.- RELATIONAL OPERAND HAS LIGICAL TYPE. -Operando relacional tiene tipo lógico - -El compilador se encontró en una expresión relacional un operador de tipo - lógico.

158.- CHARACTER VS. ARITHMETIC RELATION. -Se encontró una relación aplicable a variable character en donde se esperaba una relación aritmética.

159.- ILLEGAL RELATION FOR COMPLEX OPERANDS. - Relación ilegal para operandos complejos -Hay unas variables complejas relacionadas con un operador que no es - aplicable.

160.- OPERAND OF .NOT. NOT LOGICAL. -Un operador .NOT. está operando sobre una variable que no es del tipo LOGICAL.

161.- IMPROPER STRING DESIGNATOR. -El compilador se encontro una expresión como las que se emplean para extraer segmentos de variables tipo CHARACTER, mal utilizada.

162.- COMPLEX POWER. -El compilador se encontró un número elevado a una potencia - compleja.

163.- COMPLEX BASE TO NON INTEGER POWER. -Hay una potencia no entera de un número - complejo.

164.- STRING EXPRESSION IN PARENTHESIS. -El compilador se econtró una variable tipo CHARACTER ente paréntesis y esto solo se aplica a variables aritméticas.

165.- PARTIAL-WORD EXCEEDS 15 BITS. -Se intenta hacer referencia a un segmento de - palabra binaria mas grande que 15 bits.

166.- IMPROPER TYPE FOR PARTIAL WORD DESIGNATOR. -Dentro de una expresión que designa un segmento de palabra hay un tipo de variable incorrecto.

167.- COMPLEX INDEX EXPRESSION. -El compilador se encontró una expresión índice de tipo complejo.

168.- COMPLEX SUBSCRIPT. -El compilador se encontró un índice de tipo complejo.

169.- RECURSIVE STATEMENT FUNCTION. -El compilador se encontró un postulado definido recursivamente y lo suprimió.

170.- SUBROUTINE MISSING ARGUMENTS. -Se esta llamando una subrutina con un número -equivocado de argumentos- Cuente el número de argumentos en la definición de la subrutina y verá que no coincide con el número de argumento en la llamada.

171.- FUNCTION MISSING ARGUMENTS. -El compilador se encontró la definición de una -función sin argumentos.

173.- MISSING SUBSCRIPT. -El compilador se encontró una referencia a un arreglo pero sin subíndices.

176.- TOO FEW ARGUMENTS. -Se encontraron menos argumentos de los que se necesitan.

177.- TOO MANY ARGUMENTS. -Se encontraron mas argumentos de los que se necesitan.

180.- NO LIMIT PARAMETER. -Hay un postulado DO que no tiene definido el límite. También puede ser un DO implícito.

181.- TERMINAL LABEL PRECEDES DO STATEMENT. -La etiqueta del postulado que remata -el DO está colocada antes que el mismo DO.

182.- IMPROPERLY NESTED DO STATEMENTS. -Hay dos o mas postulados DO mal anidados. -Su rango se está traslapando.

183.- INTEGER SIMPLE VARIABLE ESPECTED. -El compilador esperaba una variable entera sola.

184.- IMPROPER TERMINAL STATEMENT. -Un postulado no permitido está cerrando un DO. Vea el manual para la lista de los postulados que pueden rematar un DO.

185.- UNDECLARED ARRAY NAME. -El compilador se encontró una variable usada como -- arreglo pero que no ha sido declarada como tal.

186.- LEFT-HAND IS FUNCTION OR SUBROUTINE. -El lado izquierdo de una asignación tiene una función o una subrutina en vez de una variable o un arreglo.

188.- RIGHT AND LEFT-HAND TYPES INCOMPATIBLE. -El compilador ha encontrado tipos de variables o expresiones que son incompatibles en ambos lados de una signación.

191.- CHARACTER STATEMENT FUNCTION. -Función de proposición de tipo "Character" no está permitida. El compilador la ignora.

194.- UNABLE TO CLASSIFY GOTO. -(Incapaz de clasificar GOTO) -Error de sintaxis en un GOTO que lo hace aparecer muy extraño.

197.- EXPECTED LOGICAL EXPRESSION. -(Esperaba expresión lógica).

198.- IMPROPER LOGICAL CLAUSE. (Cláusula lógica impropia) -El compilador ignora toda la proposición.

199.- IMPROPER DEPENDENT STATEMENT. -(Proposición dependiente impropia) -El compilador encontró una cláusula invalida a la derecha de un IF lógico (otro IF o un DO).

200.- ALTERNATE RETURN IN NON-SUBROUTINE. -("RETURN" en una nó-subrrutina) -La -proposición Return solo es válida en subrrutinas o funciones.

202.- SYMBOL NOT SUBROUTINE NAME. -(Símbolo no es nombre de subrrutina). -El compilador esperaba el nombre de una subrutina y encontró otra cosa.

203.- EXPRESSION IN INPUT LIST. -(Expresión en lista de entrada de datos. -El compilador encontró una expresión en un Read, lo cual no tiene sentido.

204.- IMPROPER I/O LIST ITEM. -Hay un elemento extraño en una lista de entrada o de salida. El compilador elimina la proposición.

205.- EXPECTED I/O LIST. -(Esperaba lista de entrada/salid). -Lista de entrada/salida ausente. La proposición se borra.

206.- IMPROPER UNIT REFERENTE. -(Referencia de unidad impropia). -El número de unidad en un Read o Write debe ser una variable o constante entera.

207.- EXPECTED CHARACTER VARIABLE. -(Esperaba variable de caracteres).

208.- EXPECTED FORMAT REFERENCE. -(Esperaba referencia a formato) -El número de formato es erróneo o ausente en un READ o WRITE.

212.- DATA SPACE OVERFLOW. -(Sobreflujo del espacio de datos) -El programa necesita demasiadas variables o arreglos muy grandes. El número máximo de elementos de arreglo reales y enteros no debe exceder de 8000 aproximadamente. Disminuya el tamaño de sus arreglos.

215.- EXPECTED AN INTEGER OR STRING. -(Esperaba entero o tira de caracteres) -Un - STOP solo puede estar seguido de un entero o de una tira, o de nada.

224.- INCORRECT DIGIT. -(Digito incorrecto) -Solo es valida INTEGER*2 ó INTEGER*4, se supone INTEGER*2.

226.- REFERENCED VARIABLE NOT DEFINED. -(Variable no definida) -La variable que se lista en el mensaje no ha sido inicializada.

230.- ITEM IN DATA REDUNDANTLY INITIALIZED. -(Variable en DATA inicializada en forma redundante). - La misma variable no puede aparecer mas de una vez en la - proposición DATA.

233.- COMPILER ERROR. -(Error del compilador) -Acuda con un profesor rápidamente.

236.- MORE THAN ONE ALTER BLOCK ACTIVE. -(Mas de un bloque exterior activo) -El - compilador encontró proposiciones ejecutables despues de un END. Posiblemente el END sale sobrando.

Ejm:
write(5,5)
2 format('Hola',10x,'Avión',TR4,'ten')

4) / : Salto de renglón.

5) \ : Lee o escribe en el mismo renglón.

- Descripción de enteros.

3 format(1, I3, 10x, I3, 1x, I3)

enteros
I3: De 3 Dígitos

- Lee 3 enteros (I3) en el mismo renglón, saltando 10 espacios

Ejercicio: escribir x | sen x

L. o C.
Numéricos
I enteros
F, E, D, G reales

L lógicos o Booleanos
A caracteres.

Ejm:
Read(5,10) N
2 format(5x, I3)

I W → número
tipo
W = width

Ejm: read(5,10) N1, N2, N3
10 format(3I3)

= F =
read(5,10) sen x
2 Format(F 4.2)
- El truncamiento es de 4 con 2 decimales
1.99
1234

= E =
2 Format (E 4.2)
variables con exponente
a.99 E±999

= L =
read(5,10) R1:R2
2 Format(L1)

= A = A W
read(5,1) RespP
Format (A1)