CNEA - NT 6/82

REPUBLICA ARGENTINA
COMISION NACIONAL DE ENERGIA ATOMICA
DEPENDIENTE DE LA PRESIDENCIA DE LA NACION

GUIA DE PROGRAMAS FORTRAN PARA EL USO Y TRANSMISION A RUTINAS DE CONSTANTES Y VARIABLES LITERALES EN SISTEMA OPERATIVO IBM/370

ALICIA DIAZ ROMERO

BUENOS AIRES
1982

CNEA - NT 6/82

REPUBLICA ARGENTINA
COMISION NACIONAL DE ENERGIA ATOMICA
DEPENDIENTE DE LA PRESIDENCIA DE LA NACION

GUIA DE PROGRAMAS FORTRAN PARA EL USO Y TRANSMISION A RUTINAS DE CONSTANTES Y VARIABLES LITERALES EN SISTEMA OPERATIVO IBM/370

ALICIA DIAZ ROMERO

BUENOS AIRES
1982

I N D I C E

ADVERTENCIA A LOS USUARIÓS

Los índices deben ser utilizados en el orden en que han sido preparados. En el INDICE I se selecciona el Capítulo, en el INDICE II se elige el Programa y en el INDICE III se accede al Caso que se requiere.

I N T R O D U C C I O N

El objeto de este trabajo es ejercitar al lector en el empleo de literales como un medio de acrecentar la eficiencia del lenguaje FORTRAN IV, muy difundido para realizar cálculos científicos, en el ámbito del Departamento de Física de la CNEA. No es un manual que enseñe este lenguaje, por el contrario, se supone que el usuario de la Guía tiene por lo menos un conocimiento discreto del mismo y sólo necesita ejemplos sobre dos puntos claves: cómo se comparan las tiras de caracteres entre sí y cómo se transmiten a rutinas.

Cómo se comparan. Conviene insistir en que las tiras de caracteres que se van a comparar tienen que ser del mismo tipo siguiendo las reglas de declaración implícita según la letra del comienzo, de acuerdo a la sintaxis del lenguaje FORTRAN. También se pueden equiparar, cualquiera sea la primera letra, por declaración explícita REAL o INTEGER. Además ambas tiras deben tener la misma cantidad de octetos que será implícitamente de 4 o si no, explícitamente de 2 o de 8 octetos, según convenga en el programa.

Cómo se transmiten. Cada idea que se propone sobre la posible utilización de literales va acompañada del correspondiente programa para llevarla a la práctica, presentado de dos maneras: a) las tiras de caracteres ingresan por formato de lectura y b) las tiras están definidas en sentencias DATA. Cada uno de estos tópicos está desglosado a su vez en los casos típicos en que no hay rutinas a las cuales transmitir los literales o si las hay, la transmisión se realiza por medio de una sentencia COMMON o se incorporan como argumentos en la sentencia CALL que las invoca.

Aunque esta estructura se mantiene a lo largo de toda la Guía (salvo en los dos últimos capítulos que se refieren a la búsqueda de caracteres), se ha procurado, por una parte que las dificultades aparezcan en orden creciente de complejidad a medida que se desarrollan los distintos temas; por otro lado se ha sacrificado en más de una oportunidad la optimización de los programas en pro de su claridad, dándole más importancia a la didáctica que a la técnica ya que de todas maneras éstos no van a ser computados tal cual se publican. En efecto, los sencillos cálculos que se realizan en ellos no significan absolutamente nada ni conducen a ninguna parte; son solamente a modo de un armazón, provisto de cierta lógica, destinado a sustentar el manejo de literales que se desea exponer.

Conviene recordar que casi todos los casos considerados tienen sentido en grandes programas de diez mil o más tarjetas, pués en los pequeños muchas veces resulta obvio lo que en principio se facilitaría enormemente usando tiras de caracteres.

En el Capítulo IV en el cual se dan ejemplos de formatos variables de lectura, se hace notar que la especificación de estos tiene que ser un vector, aún cuando su dimensión sea 1.

En los Capítulos V y VI relativos a formatos variables de escritura puede llamar la atención, en el primero el complicado uso de los apóstrofos y en el segundo el, más complicado aún, uso del código Hollerith, en los respectivos Casos 4, 5 y 6, en que se utilizan sentencias DATA. Aparte de razones de metodología que aconsejan la inclusión de estos casos, se los suele encontrar de tanto en tanto en programas en uso, lo que justifica su comentario.

En el Capítulo IX se incluyen programas FORTRAN para buscar tiras de caracteres en otros programas escritos con anterioridad, generalmente a los efectos de distinguir una versión de otra similar. Estos últimos, que desempeñan el papel de juegos de datos, pueden estar perforados en tarjetas, residir en bibliotecas de simbólicos o estar grabados en cintas magnéticas, teniéndose en cuenta en los ejemplos ofrecidos cada uno de estos tres soportes físicos.

En el Capítulo X se ha agregado el utilitario OSLISTA, escrito en 1971 por el Sr. Jorge H. Vattuone, Analista de Sistemas del Centro Unico de Procesamiento Electrónico de Datos (CUPED); este utilitario está disponible en el Centro de Cómputos de la CNEA y es extraordinariamente rápido en la búsqueda de grupos de caracteres.

Aunque siempre se menciona la "tarjeta" como entrada de datos en Sistema Operativo, queda sobreentendido que esta palabra puede ser reemplazada por "imagen de tarjeta" cuando el programa se envía a la computadora desde una máquina virtual o se utiliza el Conversational Monitor System (CMS) en vez del OS.

Para facilitar el uso de la Guía, cada programa va acompañado de una sucinta descripción de su estructura al principio del Capítulo al cual pertenece y cada uno de los elementos que la componen ha sido marcado, no solamente con el número de orden del CASO que representa, sino también con el número del PROGRAMA y del CAPITULO que integra y está precedido por un breve resumen de su finalidad.

Se recomienda un vistazo general de la Guía antes de proceder a su utilización. La autora espera haber puesto en claro cómo se salvan las principales dificultades en la manipulación de literales: por lo menos éste ha sido su deseo.

Todos los programas que aquí figuran han sido probados en la práctica.

Buenos Aires, abril de 1982

A G R A D E C I M I E N T O S

Al Dr. Ernesto E. Maqueda del Departamento de Física de la CNEA debo agradecer valiosas enseñanzas e inapreciable ayuda en la solución de las dificultades que se presentaron durante la realización de mi trabajo.

A la Dra. Silvia L. Reich y al Lic. Aníbal O. Gattone del Departamento de Física de la CNEA deseo manifestarles mi reconocimiento por su interés en mi labor y sus precisas respuestas a mis preguntas.

Al Lic. Julio C. Durán del Departamento de Prospectiva y Estudios Especiales de la CNEA le agradezco su eficiente colaboración en la redacción de los PROGRAMAS 17, 18 y 19 DEL CAPITULO VIII, relativo a la conversión de caracteres en números.

Mi reconocimiento a las personas aquí nombradas no implica, en modo alguno, asignarles responsabilidad por los errores que se puedan haber deslizado en la presente Guía.

B I B L I O G R A F I A

1. Systems OS/VS1
   Job Control Language Reference

2. IBM Systems Reference Library
   OS Utilities

3. IBM System/360 and 370
   FORTRAN IV Language

4. IBM System/360 and 370 Operating System
   FORTRAN IV (G and H) Programmer's Guide

5. McCracken Daniel D.
   A Guide to FORTRAN IV Programming

6. Jaffe Richard M.
   A Clear Introduction to FORTRAN IV

7. Díaz Romero A.
   CNEA - NT. 33/78

I N D I C E I: PLAN GENERAL DE LA GUIA

I N D I C E II. TEMARIO DE CADA CAPITULO.

I N D I C E III: TEMARIO DE CADA PROGRAMA.

PROGRAMA 5. Un programa tiene numerosos juegos de datos secuenciales precedidos cada uno por una tarjeta con un título de 4 caracteres. Se desea

CAPITULO I

IDENTIFICACION DE LA RAMA POR LA CUAL DERIVO EL PROGRAMA

PROGRAMA 1.

OBJETO: Se colocan marcas literales para señalar el camino por el cual avanza el programa de acuerdo a las decisiones adoptadas por sentencias IF. Al llegar a un cierto punto estas marcas se comparan y según el resultado se efectuarán o no nuevos cálculos.

CASOS CONSIDERADOS

A) Los datos literales ingresan al programa principal por medio de un formato de lectura, cada uno a una variable.

CASO 1. El programa no tiene rutinas a las cuales se transmitan las variables literales.
CASO 2. Las variables literales se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. Las variables literales se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

B) Los datos literales están definidos por medio de una sentencia DATA para cada uno o en el programa principal o en una subrutina BLOCK DATA.

CASO 4. El programa no tiene rutinas a las cuales se transmitan las constantes literales.
CASO 5. Las constantes literales están incluídas en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. Las constantes literales se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 y 4.

Los datos literales son FLG1 y FLG2 que ingresan a las variables UNO y DOS por formato de lectura (CASO 1) o bien están definidos como constantes en las sentencias DATA de nombres UNO y DOS (CASO 4).

Con A=UNO se señala la entrada al programa. Si la sentencia

```
IF(VAT.LE.3.2) GO TO 3
```

no se cumpliera, se señalará la entrada a la marca 2 con B=UNO; si se cumpliera se señalará la entrada en la marca 3 con B=DOS. Cuando el programa llega a la marca 4, la sentencia

```
IF(A.EQ.B) VAT=VAT-Z
```

basada en la comparación de las tiras de caracteres, producirá o no una variación

en el cálculo.

Toda vez que se realizare una comparación de tiras de caracteres en una sentencia IF, las variables involucradas en la comparación deberán ser del mismo tipo. Las variables UNO y DOS son ambas REAL*4 por convención implícita.

CAP.I,PROGR.1,CASO 1. Se señala el camino por el cual avanzará el programa. Los datos literales FLG1 y FLG2 ingresan al programa principal por medio de un formato de lectura, a las variables UNO y DOS respectivamente. El programa no tiene rutinas a las cuales se transmitan estas variables.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) UNO,DOS
      READ(5,102) V,W,X,Y,Z
      A=UNO
      VAT=V+Y-W
      IF(VAT.LE.3.2) GO TO 3
    2 B=UNO
      VAT=VAT*Z
      GO TO 4
    3 B=DOS
    4 VAT=VAT*(X-Y)
      IF(A.EQ.B) VAT=VAT-Z
      WRITE(6,101) VAT
      STOP
  100 FORMAT(2A4)
  101 FORMAT(1X,'VAT=',F5.1)
  102 FORMAT(5F5.1)
      END
/*
//GO.SYSIN DD *
FLG1FLG2
 -3.7 -0.5  6.3  5.7 -7.3
/*
//
```

CAP.I,PROGR.1,CASO 2. Se señala el camino por el cual avanzará el programa. Los datos literales FLG1 y FLG2 ingresan al programa principal por medio de un formato de lectura, a las variables UNO y DOS respectivamente, las cuales se transmiten a la SUBROUTINE CALCUL por una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/CARAC/UNO,DOS
      READ(5,100) UNO,DOS
      READ(5,102) V,W,X,Y,Z
      CALL CALCUL(V,W,X,Y,Z)
      STOP
```

```
  100 FORMAT(2A4)
  102 FORMAT(5F5.1)
      END
C
      SUBROUTINE CALCUL(V,W,X,Y,Z)
      COMMON/CARAC/UNO,DOS
      A=UNO
      VAT=V+Y-W
      IF(VAT.LE.3.2) GO TO 3
    2 B=UNO
      VAT=VAT*Z
      GO TO 4
    3 B=DOS
    4 VAT=VAT*(X-Y)
      IF(A.EQ.B) VAT=VAT-Z
      WRITE(6,101) VAT
      RETURN
  101 FORMAT(1X,'VAT=',F5.1)
      END
/*
//GO.SYSIN DD *
FLG1FLG2
 -3.7 -0.5  6.3  5.7 -7.3
/*
//
```

CAP.I,PROGR.1,CASO 3. Se señala el camino por el cual avanzará el programa. Los datos literales FLG1 y FLG2 ingresan al programa principal por medio de un formato de lectura, a las variables UNO y DOS respectivamente, las cuales se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) UNO,DOS
      READ(5,102) V,W,X,Y,Z
      CALL CALCUL(UNO,DOS,V,W,X,Y,Z)
      STOP
  100 FORMAT(2A4)
  102 FORMAT(5F5.1)
      END
C
      SUBROUTINE CALCUL(UNO,DOS,V,W,X,Y,Z)
      A=UNO
      VAT=V+Y-W
      IF(VAT.LE.3.2) GO TO 3
    2 B=UNO
      VAT=VAT*Z
      GO TO 4
    3 B=DOS
    4 VAT=VAT*(X-Y)
      IF(A.EQ.B) VAT=VAT-Z
      WRITE(6,101) VAT
      RETURN
```

```
  101 FORMAT(1X,'VAT=',F5.1)
      END
/*
//GO.SYSIN DD *
FLG1FLG2
 -3.7 -0.5  6.3  5.7 -7.3
/*
//
```

CAP.I,PROGR.1,CASO 4. Se señala el camino por el cual avanzará el programa. Los datos literales FLG1 y FLG2 están definidos en el programa principal por medio de las sentencias DATA de nombres UNO y DOS respectivamente. El programa no tiene rutinas a las cuales se transmitan las constantes literales.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA UNO/'FLG1'/
      DATA DOS/'FLG2'/
      READ(5,102) V,W,X,Y,Z
      A=UNO
      VAT=V+Y-W
      IF(VAT.LE.3.2) GO TO 3
    2 B=UNO
      VAT=VAT*Z
      GO TO 4
    3 B=DOS
    4 VAT=VAT*(X-Y)
      IF(A.EQ.B) VAT=VAT-Z
      WRITE(6,101) VAT
      STOP
  100 FORMAT(2A4)
  101 FORMAT(1X,'VAT=',F5.1)
  102 FORMAT(5F5.1)
      END
/*
//GO.SYSIN DD *
 -3.7 -0.5  6.3  5.7 -7.3
/*
//
```

CAP.I,PROGR.1,CASO 5. Se señala el camino por el cual avanzará el programa. Los datos literales FLG1 y FLG2 están definidos por medio de las sentencias DATA de nombres UNO y DOS respectivamente, incluídas en un BLOCK DATA. Desde allí se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,102) V,W,X,Y,Z
      CALL CALCUL(V,W,X,Y,Z)
      STOP
```

```
  102 FORMAT(5F5.1)
      END
C
      BLOCK DATA
      COMMON/CARAC/UNO,DOS
      DATA UNO/'FLG1'/
      DATA DOS/'FLG2'/
      END
C
      SUBROUTINE CALCUL(V,W,X,Y,Z)
      COMMON/CARAC/UNO,DOS
      A=UNO
      VAT=V+Y-W
      IF(VAT.LE.3.2) GO TO 3
    2 B=UNO
      VAT=VAT*Z
      GO TO 4
    3 B=DOS
    4 VAT=VAT*(X-Y)
      IF(A.EQ.B) VAT=VAT-Z
      WRITE(6,101) VAT
      RETURN
  100 FORMAT(2A4)
  101 FORMAT(1X,'VAT=',F5.1)
      END
/*
//GO.SYSIN DD *
 -3.7 -0.5  6.3  5.7 -7.3
/*
//
```

CAP.I,PROGR.1,CASO 6. Se señala el camino por el cual avanzará el programa. Los datos literales FLG1 y FLG2 están definidos en el programa principal por medio de sentencias DATA de nombres UNO y DOS respectivamente, las cuales se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA UNO/'FLG1'/
      DATA DOS/'FLG2'/
      READ(5,102) V,W,X,Y,Z
      CALL CALCUL(UNO,DOS,V,W,X,Y,Z)
      STOP
  102 FORMAT(5F5.1)
      END
C
      SUBROUTINE CALCUL(UNO,DOS,V,W,X,Y,Z)
      A=UNO
      VAT=V+Y-W
      IF(VAT.LE.3.2) GO TO 3
    2 B=UNO
      VAT=VAT*Z
      GO TO 4
```

```
     3 B=DOS
     4 VAT=VAT*(X-Y)
       IF(A.EQ.B) VAT=VAT-Z
       WRITE(6,101) VAT
       RETURN
   100 FORMAT(2A4)
   101 FORMAT(1X,'VAT=',F5.1)
       END
/*
//GO.SYSIN DD *
 -3.7 -0.5  6.3  5.7 -7.3
/*
//
```

PROGRAMA 2.

OBJETO: En un programa que deriva a una de tres posibles ramas que luego convergen en un punto, la colocación de marcas literales en cada una permite identificar por cuál de ellas pasó el programa, antes de iniciar una nueva serie de bifurcaciones.

CASOS CONSIDERADOS

A) Los datos literales son variables e ingresan al programa principal por medio de un formato de lectura, como elementos de un vector.

CASO 1. El programa no tiene rutinas a las cuales se transmita el vector de datos literales.
CASO 2. El vector de datos literales se transmite a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. El vector de datos literales se transmite a una rutina de cálculo como argumento en la sentencia CALL que la invoca.

B) Los datos literales son constantes y están definidos en una sola sentencia DATA o en el programa principal o en una subrutina BLOCK DATA.

CASO 4. El programa no tiene rutinas a las cuales se transmitan las constantes literales.
CASO 5. Las constantes literales están incluídas en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. Las constantes literales se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 y 4.

Los datos literales son FLG1, FLG2 y FLG3. El programa adquirirá esta información de alguna de las siguientes manera: a) los datos ingresan por formato de lectura como elementos del vector BUSCA, REAL*4, de dimensión 3 (CASO 1); b) los datos están definidos como constantes en una sola sentencia DATA de nombre BUSCA, REAL*4, de dimensión 3 (CASO 4).

Con FLAG=BUSCA(1) se señalará la entrada a la rama 1 del programa; con FLAG=BUSCA(2) y FLAG=BUSCA(3) se señalará la entrada a las ramas 2 y 3 respectivamente. En el punto de convergencia (marca 4) se compararán las tiras de caracteres y según el resultado de la misma se derivará nuevamente el programa a otras 3 ramas. En la marca 4, si por error del programador no se cumpliera ninguna de

las sentencias IF, la sentencia GO TO 8 transferirá el control del programa al STOP y la ejecución terminará. Esta precaución se ha tomado solamente, como un ejemplo, en el CASO 1.

Toda vez que se realizare una comparación de tiras de caracteres en una sentencia IF, las variables y vectores involucrados en la comparación, deberán ser del mismo tipo. La variable FLAG y el vector BUSCA son ambos REAL*4 por convención implícita.

CAP.I,PROGR.2,CASO 1. Se señala la rama por la cual pasó el programa. Los datos literales FLG1, FLG2 y FLG3 ingresan al programa principal por medio de un formato de lectura, al vector BUSCA, REAL*4 y de dimensión 3. El programa no tiene rutinas a las cuales se transmita este vector.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION BUSCA(3)
      READ(5,102) BUSCA
      READ(5,101) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 FLAG=BUSCA(1)
      A=B+C
      GO TO 4
    2 FLAG=BUSCA(2)
      A=B-C
      GO TO 4
    3 FLAG=BUSCA(3)
      A=-C-D
    4 IF(FLAG.EQ.BUSCA(1)) GO TO 5
      IF(FLAG.EQ.BUSCA(2)) GO TO 6
      IF(FLAG.EQ.BUSCA(3)) GO TO 7
      GO TO 8
    5 IF(B.GT.C.AND.C.GT.D) A=A*B
      WRITE(6,103) A
      GO TO 8
    6 IF(A.LE.3.0) A=A/B
      WRITE(6,103) A
      GO TO 8
    7 IF(B.LT.C.AND.A.GT.3.0) A=A+B/C
      WRITE(6,103) A
    8 STOP
  101 FORMAT(I3,3F7.2)
  102 FORMAT(3A4)
  103 FORMAT(1X,'A=',F7.2)
      END
/*
//GO.SYSIN DD *
FLG1FLG2FLG3
  0  -2.63  12.75 -34.21
/*
//
```

CAP.I,PROGR.2,CASO 2. Se señala la rama por la cual pasó el programa. Los datos literales FLG1,FLG2 y FLG3 ingresan al programa principal por medio de un formato de lectura, al vector BUSCA, REAL*4 y de dimensión 3, el cual se transmite a la SUBROUTINE CALCUL por una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/CARAC/BUSCA(3)
      READ(5,102) BUSCA
      READ(5,101) NUMERO,B,C,D
      CALL CALCUL(NUMERO,B,C,D)
      STOP
  101 FORMAT(I3,3F7.2)
  102 FORMAT(3A4)
      END
C
      SUBROUTINE CALCUL(NUMERO,B,C,D)
      COMMON/CARAC/BUSCA(3)
      IF(NUMERO) 1,2,3
    1 FLAG=BUSCA(1)
      A=B+C
      GO TO 4
    2 FLAG=BUSCA(2)
      A=B-C
      GO TO 4
    3 FLAG=BUSCA(3)
      A=-C-D
    4 IF(FLAG.EQ.BUSCA(1)) GO TO 5
      IF(FLAG.EQ.BUSCA(2)) GO TO 6
      IF(FLAG.EQ.BUSCA(3)) GO TO 7
    5 IF(B.GT.C.AND.C.GT.D) A=A*B
      WRITE(6,103) A
      GO TO 8
    6 IF(A.LE.3.0) A=A/B
      WRITE(6,103) A
      GO TO 8
    7 IF(B.LT.C.AND.A.GT.3.0) A=A+B/C
      WRITE(6,103) A
    8 RETURN
  103 FORMAT(1X,'A=',F7.2)
      END
/*
//GO.SYSIN DD *
FLG1FLG2FLG3
  0  -2.63  12.75 -34.21
/*
//
```

CAP.I,PROGR.2,CASO 3. Se señala la rama por la cual pasó el programa. Los datos literales FLG1,FLG2 y FLG3 ingresan al programa principal por medio de un formato de lectura, al vector BUSCA, REAL*4 y de dimensión 3, el cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION BUSCA(3)
      READ(5,102) BUSCA
      READ(5,101) NUMERO,B,C,D
      CALL CALCUL(BUSCA,NUMERO,B,C,D)
      STOP
  101 FORMAT(I3,3F7.2)
  102 FORMAT(3A4)
      END
C
      SUBROUTINE CALCUL(BUSCA,NUMERO,B,C,D)
      DIMENSION BUSCA(3)
      IF(NUMERO) 1,2,3
    1 FLAG=BUSCA(1)
      A=B+C
      GO TO 4
    2 FLAG=BUSCA(2)
      A=B-C
      GO TO 4
    3 FLAG=BUSCA(3)
      A=-C-D
    4 IF(FLAG.EQ.BUSCA(1)) GO TO 5
      IF(FLAG.EQ.BUSCA(2)) GO TO 6
      IF(FLAG.EQ.BUSCA(3)) GO TO 7
    5 IF(B.GT.C.AND.C.GT.D) A=A*B
      WRITE(6,103) A
      GO TO 8
    6 IF(A.LE.3.0) A=A/B
      WRITE(6,103) A
      GO TO 8
    7 IF(B.LT.C.AND.A.GT.3.0) A=A+B/C
      WRITE(6,103) A
    8 RETURN
  103 FORMAT(1X,'A=',F7.2)
      END
/*
//GO.SYSIN DD *
FLG1FLG2FLG3
  0  -2.63  12.75 -34.21
/*
//
```

CAP.I,PROGR.2,CASO 4. Se señala la rama por la cual pasó el programa. Los datos literales FLG1,FLG2 y FLG3 están definidos en el programa principal por medio de una sentencia DATA de nombre BUSCA, REAL*4 y de dimensión 3. El programa no tiene rutinas a las cuales se transmita este DATA.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION BUSCA(3)
      DATA BUSCA/'FLG1','FLG2','FLG3'/
```

```
      READ(5,101) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 FLAG=BUSCA(1)
      A=B+C
      GO TO 4
    2 FLAG=BUSCA(2)
      A=B-C
      GO TO 4
    3 FLAG=BUSCA(3)
      A=-C-D
    4 IF(FLAG.EQ.BUSCA(1)) GO TO 5
      IF(FLAG.EQ.BUSCA(2)) GO TO 6
      IF(FLAG.EQ.BUSCA(3)) GO TO 7
    5 IF(B.GT.C.AND.C.GT.D) A=A*B
      WRITE(6,103) A
      GO TO 8
    6 IF(A.LE.3.0) A=A/B
      WRITE(6,103) A
      GO TO 8
    7 IF(B.LT.C.AND.A.GT.3.0) A=A+B/C
      WRITE(6,103) A
    8 STOP
  101 FORMAT(I3,3F7.2)
  103 FORMAT(1X,'A=',F7.2)
      END
/*
//GO.SYSIN DD *
  0  -2.63  12.75 -34.21
/*
//
```

CAP.I,PROGR.2,CASO 5. Se señala la rama por la cual pasó el programa. Los datos literales FLG1,FLG2 y FLG3 están definidos en un DATA de nombre BUSCA, REAL*4 y de dimensión 3, incluído en un BLOCK DATA. Desde allí se transmite a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,101) NUMERO,B,C,D
      CALL CALCUL(NUMERO,B,C,D)
      STOP
  101 FORMAT(I3,3F7.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/BUSCA(3)
      DATA BUSCA/'FLG1','FLG2','FLG3'/
      END
C
      SUBROUTINE CALCUL(NUMERO,B,C,D)
      COMMON/CARAC/BUSCA(3)
      IF(NUMERO) 1,2,3
    1 FLAG=BUSCA(1)
```

```
      A=B+C
      GO TO 4
    2 FLAG=BUSCA(2)
      A=B-C
      GO TO 4
    3 FLAG=BUSCA(3)
      A=-C-D
    4 IF(FLAG.EQ.BUSCA(1)) GO TO 5
      IF(FLAG.EQ.BUSCA(2)) GO TO 6
      IF(FLAG.EQ.BUSCA(3)) GO TO 7
    5 IF(B.GT.C.AND.C.GT.D) A=A*B
      WRITE(6,103) A
      GO TO 8
    6 IF(A.LE.3.0) A=A/B
      WRITE(6,103) A
      GO TO 8
    7 IF(B.LT.C.AND.A.GT.3.0) A=A+B/C
      WRITE(6,103) A
    8 RETURN
  103 FORMAT(1X,'A=',F7.2)
      END
/*
//GO.SYSIN DD *
  0  -2.63  12.75 -34.21
/*
//
```

CAP.I,PROGR.2,CASO 6. Se señala la rama por la cual pasó el programa. Los datos literales FLG1,FLG2 y FLG3 están definidos en un DATA de nombre BUSCA, REAL*4 y de dimensión 3, el cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION BUSCA(3)
      DATA BUSCA/'FLG1','FLG2','FLG3'/
      READ(5,101) NUMERO,B,C,D
      CALL CALCUL(BUSCA,NUMERO,B,C,D)
      STOP
  101 FORMAT(I3,3F7.2)
      END
C
      SUBROUTINE CALCUL(BUSCA,NUMERO,B,C,D)
      DIMENSION BUSCA(3)
      IF(NUMERO) 1,2,3
    1 FLAG=BUSCA(1)
      A=B+C
      GO TO 4
    2 FLAG=BUSCA(2)
      A=B-C
      GO TO 4
    3 FLAG=BUSCA(3)
      A=-C-D
```

```
    4 IF(FLAG.EQ.BUSCA(1)) GO TO 5
      IF(FLAG.EQ.BUSCA(2)) GO TO 6
      IF(FLAG.EQ.BUSCA(3)) GO TO 7
    5 IF(B.GT.C.AND.C.GT.D) A=A*B
      WRITE(6,103) A
      GO TO 8
    6 IF(A.LE.3.0) A=A/B
      WRITE(6,103) A
      GO TO 8
    7 IF(B.LT.C.AND.A.GT.3.0) A=A+B/C
      WRITE(6,103) A
    8 RETURN
  103 FORMAT(1X,'A=',F7.2)
      END
/*
//GO.SYSIN DD *
  0  -2.63  12.75 -34.21
/*
//
```

CAPITULO II

TRANSFERENCIA DEL CONTROL DEL PROGRAMA

PROGRAMA 3.

OBJETO: El programa tiene varias posibles entradas que corresponden a distintos cálculos. Según una palabra clave que se comparará con cada una de un grupo de palabras definidas en una sentencia DATA que desempeña el papel de un monitor, el control se transferirá a alguna de dichas marcas.

CASOS CONSIDERADOS

A) La palabra clave ingresa al programa por medio de un formato de lectura.

CASO 1. La palabra clave ingresa al programa principal. El DATA monitor está incluído en ese mismo programa y allí se compararán.

CASO 2. La palabra clave ingresa al programa principal; desde allí se transmite por medio de una sentencia COMMON a la rutina en la cual está incluído el DATA monitor y allí se compararán.

CASO 3. La palabra clave ingresa al programa principal; se transmite como argumento en la sentencia CALL que la invoca, a la rutina en la cual está incluído el DATA monitor y allí se compararán.

CASO 4. La palabra clave ingresa al programa principal y se transmite como argumento en la sentencia CALL que la invoca, a la rutina en la cual se comparará con el DATA monitor. Este está incluído en un BLOCK DATA y se transmite a dicha rutina por medio de una sentencia COMMON.

CASO 5. La palabra clave ingresa al programa principal. El DATA monitor está incluído en un BLOCK DATA. Ambos se transmiten por medio de una sentencia COMMON a la rutina en la cual se compararán.

CASO 6. La palabra clave ingresa a una rutina. El DATA monitor está incluído en un BLOCK DATA: se transmite a dicha rutina por medio de una sentencia COMMON y allí se compararán.

B) La palabra clave está definida en el programa por medio de un DATA.

CASO 7. El DATA de la palabra clave está incluído en el programa principal. El DATA monitor está incluído en ese mismo programa y allí se compararán.

CASO 8. El DATA de la palabra clave está incluído en un BLOCK DATA; se transmite por medio de una sentencia COMMON a la rutina en la cual está incluído el DATA monitor y allí se compararán.

CASO 9. El DATA de la palabra clave está incluído en el programa principal; se transmite como argumento en la sentencia CALL que la invoca, a la rutina en la cual está incluído el DATA monitor y allí se compararán.

CASO 10. El DATA de la palabra clave está incluído en el programa principal; se transmite como argumento en la sentencia CALL que la invoca, a la rutina en la cual se comparará con el DATA monitor. Este está incluído en un BLOCK DATA y se transmite a esta rutina por medio de una sentencia COMMON.

CASO 11. El DATA de la palabra clave y el DATA monitor están incluídos en el mismo BLOCK DATA. Ambos se transmiten, por medio de una sentencia COMMON, a la rutina en la cual se compararán.

CASO 12. El DATA de la palabra clave está incluído en una rutina. El DATA monitor está incluído en un BLOCK DATA; se transmite a dicha rutina por medio de una sentencia COMMON y allí se compararán.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 y 7.

El dato literal es la palabra RAIZ que ingresa al programa por formato de lectura, a la variable CLAVE, REAL*4,(CASO 1) o bien está definida en la sentencia DATA de nombre CLAVE, REAL*4, (CASO 7). Esta tira de caracteres se comparará mediante el DO 10 con cada una de las palabras del DATA monitor de nombre AMARCA, REAL*4 y de dimensión 7. Cuando encuentre la igualdad ramificará hacia una sentencia GO TO computada (marca 11) que hace las veces de distribuidora y transferirá el control del programa a la marca que corresponda. CLAVE y AMARCA podrán ser comparados en una sentencia IF por ser ambos del mismo tipo.

CAP.II,PROGR.3,CASO 1. Se transfiere el control del programa a una marca. La palabra RAIZ ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE. El DATA monitor de nombre AMARCA está incluído en ese mismo programa y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1  EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION AMARCA(7)
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      READ(5,100) CLAVE
      READ(5,102) A,B
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4,5,6,7),I
    1 C=A+B
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
    4 C=A/B
      GO TO 9
```

```
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 STOP
  100 FORMAT(A4)
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
RAIZ
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 2. Se transfiere el control del programa a una marca. La palabra RAIZ ingresa al programa principal,por medio de un formato de lectura, a la variable CLAVE, la cual se transmite a la SUBROUTINE CALCUL por una sentencia COMMON. El DATA monitor de nombre AMARCA está incluído en dicha rutina y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/CARAC/CLAVE
      READ(5,100) CLAVE
      READ(5,102) A,B
      CALL CALCUL(A,B)
      STOP
  100 FORMAT(A4)
  102 FORMAT(2F6.2)
      END
C
      SUBROUTINE CALCUL(A,B)
      DIMENSION AMARCA(7)
      COMMON/CARAC/CLAVE
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4,5,6,7),I
    1 C=A+B
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
```

```
      GO TO 9
    4 C=A/B
      GO TO 9
    5 C= SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 RETURN
  101 FORMAT(1X,E14.6)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
RAIZ
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 3. Se transfiere el control del programa a una marca. La palabra RAIZ ingresa al programa principal, por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca. El DATA monitor de nombre AMARCA está incluído en dicha rutina y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) CLAVE
      READ(5,102) A,B
      CALL CALCUL(CLAVE,A,B)
      STOP
  100 FORMAT(A4)
  102 FORMAT(2F6.2)
      END
C
      SUBROUTINE CALCUL(CLAVE,A,B)
      DIMENSION AMARCA(7)
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4,5,6,7),I
    1 C=A+B
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
```

```
    4 C=A/B
      GO TO 9
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 RETURN
  101 FORMAT(1X,E14.6)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
RAIZ
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 4. Se transfiere el control del programa a una marca. La palabra RAIZ ingresa al programa principal, por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca. El DATA monitor de nombre AMARCA está incluído en un BLOCK DATA y se transmite, por medio de una sentencia COMMON, a dicha rutina en donde se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) CLAVE
      READ(5,102) A,B
      CALL CALCUL(CLAVE,A,B)
      STOP
  100 FORMAT(A4)
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/AMARCA(7)
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      END
C
      SUBROUTINE CALCUL(CLAVE,A,B)
      COMMON/CARAC/AMARCA(7)
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4,5,6,7),I
    1 C=A+B
```

```
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
    4 C=A/B
      GO TO 9
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 RETURN
  101 FORMAT(1X,E14.6)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
RAIZ
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 5. Se transfiere el control del programa a una marca. La palabra RAIZ ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE CALCUL por una sentencia COMMON. El DATA monitor de nombre AMARCA está incluído en un BLOCK DATA y se transmite, por la misma sentencia COMMON, a dicha rutina en donde se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/CARAC/CLAVE,AMARCA(7)
      READ(5,100) CLAVE
      READ(5,102) A,B
      CALL CALCUL(A,B)
      STOP
  100 FORMAT(A4)
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/CLAVE,AMARCA(7)
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      END
C
      SUBROUTINE CALCUL(A,B)
      COMMON/CARAC/CLAVE,AMARCA(7)
```

```
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4,5,6,7),I
    1 C=A+B
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
    4 C=A/B
      GO TO 9
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 RETURN
  101 FORMAT(1X,E14.6)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
RAIZ
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 6. Se transfiere el control del programa a una marca. La palabra RAIZ ingresa a la SUBROUTINE CALCUL por medio de un formato de lectura, a la variable CLAVE. El DATA monitor de nombre AMARCA está incluído en un BLOCK DATA y se transmite, por una sentencia COMMON, a dicha rutina en donde se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,102) A,B
      CALL CALCUL(A,B)
      STOP
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/AMARCA(7)
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      END
C
      SUBROUTINE CALCUL(A,B)
```

```
      COMMON/CARAC/AMARCA(7)
      READ(5,100) CLAVE
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4,5,6,7),I
    1 C=A+B
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
    4 C=A/B
      GO TO 9
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 RETURN
  100 FORMAT(A4)
  101 FORMAT(1X,E14.6)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
 23.25 12.35
RAIZ
/*
//
```

CAP.II,PROGR.3,CASO 7. Se transfiere el control del programa a una marca. La palabra RAIZ está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE. El DATA monitor de nombre AMARCA está incluído en ese mismo programa y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION AMARCA(7)
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      DATA CLAVE/'RAIZ'/
      READ(5,102) A,B
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO (1,2,3,4,5,6,7),I
    1 C=A+B
```

```
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
    4 C=A/B
      GO TO 9
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 STOP
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR'/
      END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 8. Se transfiere el control del programa a una marca. La palabra RAIZ está definida en una sentencia DATA de nombre CLAVE incluída en un BLOCK DATA. Desde allí se transmite, por una sentencia COMMON a la SUBROUTINE CALCUL. El DATA monitor de nombre AMARCA está incluído en dicha rutina y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,102) A,B
      CALL CALCUL(A,B)
      STOP
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/CLAVE
      DATA CLAVE/'RAIZ'/
      END
C
      SUBROUTINE CALCUL(A,B)
      DIMENSION AMARCA(7)
      COMMON/CARAC/CLAVE
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
```

```
    10 CONTINUE
       GO TO 8
    11 GO TO(1,2,3,4,5,6,7),I
     1 C=A+B
       GO TO 9
     2 C=A-B
       GO TO 9
     3 C=A*B
       GO TO 9
     4 C=A/B
       GO TO 9
     5 C=SQRT(A)
       GO TO 9
     6 C=EXP(A)
       GO TO 9
     7 C=ALOG(A)
       GO TO 9
     8 WRITE(6,103)
       GO TO 12
     9 WRITE(6,101) C
    12 RETURN
   101 FORMAT(1X,E14.6)
   103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
       END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 9. Se transfiere el control del programa a una marca. La palabra RAIZ está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE. Desde allí se transmite, como argumento en la sentencia CALL que la invoca, a la SUBROUTINE CALCUL. El DATA monitor de nombre AMARCA está incluído en dicha rutina y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA CLAVE/'RAIZ'/
      READ(5,102) A,B
      CALL CALCUL(CLAVE,A,B)
      STOP
  102 FORMAT(2F6.2)
      END
C
      SUBROUTINE CALCUL(CLAVE,A,B)
      DIMENSION AMARCA(7)
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4,5,6,7),I
```

```
    1 C=A+B
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
    4 C=A/B
      GO TO 9
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 RETURN
  101 FORMAT(1X,E14.6)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 10. Se transfiere el control del programa a una marca. La palabra RAIZ está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE. Desde allí se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca. El DATA monitor de nombre AMARCA está incluído en un BLOCK DATA y se transmite, por medio de una sentencia COMMON, a dicha rutina en donde se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA CLAVE /'RAIZ'/
      READ(5,102) A,B
      CALL CALCUL(CLAVE,A,B)
      STOP
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/AMARCA(7)
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      END
C
      SUBROUTINE CALCUL(CLAVE,A,B)
      COMMON/CARAC/ AMARCA(7)
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
```

```
   10 CONTINUE
      GO TO 8
   11 GO TO (1,2,3,4,5,6,7),I
    1 C=A+B
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
    4 C=A/B
      GO TO 9
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 RETURN
  101 FORMAT(1X,E14.6)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 11. Se transfiere el control del programa a una marca. La palabra RAIZ está definida por medio de una sentencia DATA de nombre CLAVE incluída en un BLOCK DATA. El DATA monitor de nombre AMARCA está incluído en ese mismo BLOCK DATA. Ambos se transmiten por la misma sentencia COMMON a la SUBROUTINE CALCUL en la cual se compararán.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,102) A,B
      CALL CALCUL(A,B)
      STOP
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/AMARCA(7),CLAVE
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG'/
      DATA CLAVE/'RAIZ'/
      END
C
      SUBROUTINE CALCUL(A,B)
      COMMON/CARAC/AMARCA(7),CLAVE
      DO 10 I=1,7
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
```

```
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4,5,6,7),I
    1 C=A+B
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
    4 C=A/B
      GO TO 9
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 RETURN
  101 FORMAT(1X,E14.6)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAP.II,PROGR.3,CASO 12. Se transfiere el control del programa a una marca. La palabra RAIZ está definida por medio de una sentencia DATA de nombre CLAVE en la SUBROUTINE CALCUL. El DATA monitor de nombre AMARCA está incluído en un BLOCK DATA y se transmite, por una sentencia COMMON, a dicha rutina en donde se comparará con la palabra clave.

```
// .... JOB ......
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,102) A,B
      CALL CALCUL(A,B)
      STOP
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/AMARCA(7)
      DATA AMARCA/'SUMA','REST','MULT','DIVI','RAIZ','EXP ','ALOG')
      END
C
      SUBROUTINE CALCUL(A,B)
      COMMON/CARAC/AMARCA(7)
      DATA CLAVE/'RAIZ'/
      DO 10 I=1,7
```

```
      IF(CLAVE.EQ.AMARCA(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4,5,6,7),I
    1 C=A+B
      GO TO 9
    2 C=A-B
      GO TO 9
    3 C=A*B
      GO TO 9
    4 C=A/B
      GO TO 9
    5 C=SQRT(A)
      GO TO 9
    6 C=EXP(A)
      GO TO 9
    7 C=ALOG(A)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 RETURN
  101 FORMAT(1X,E14.6)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

PROGRAMA 4.

OBJETO: El programa tiene varias rutinas que efectúan distintos cálculos. Según una palabra clave que se comparará con cada una de un grupo de palabras definidas en una sentencia DATA, que desempeña el papel de un monitor, el control se transferirá a alguna de las rutinas de cálculo. Cuando la comparación entre la palabra clave y el DATA monitor no se realizara en el programa principal sino en una rutina, ésta será llamada rutina analizadora.

CASOS CONSIDERADOS

A) La palabra clave ingresa al programa por medio de un formato de lectura.

CASO 1. La palabra clave ingresa al programa principal. El DATA monitor está incluído en ese mismo programa y allí se compararán.
CASO 2. La palabra clave ingresa al programa principal; desde allí se transmite por medio de una sentencia COMMON a la rutina analizadora en la cual está incluído el DATA monitor y allí se compararán.
CASO 3. La palabra clave ingresa al programa principal; se transmite como argumento en la sentencia CALL que la invoca, a la rutina analizadora en la cual está incluído el DATA monitor y allí se compararán.
CASO 4. La palabra clave ingresa al programa principal y se transmite como argumento en la sentencia CALL que la invoca, a la rutina analizadora en la cual se comparará con el DATA monitor. Este está incluído en un BLOCK DATA y se transmite a dicha rutina por medio de una sentencia COMMON.
CASO 5. La palabra clave ingresa al programa principal. El DATA monitor está incluído en un BLOCK DATA. Ambos se transmiten, por medio de una sentencia COMMON, a la rutina analizadora en la cual se compararán.
CASO 6. La palabra clave ingresa a la rutina analizadora. El DATA monitor está incluído en un BLOCK DATA; se transmite a dicha rutina por medio de una sentencia COMMON y allí se compararán.

B) La palabra clave está definida en el programa por medio de un DATA.

CASO 7. El DATA de la palabra clave está incluído en el programa principal. El DATA monitor está incluído en ese mismo programa y allí se compararán.
CASO 8. El DATA de la palabra clave está incluído en un BLOCK DATA; se transmite por medio de una sentencia COMMON a la rutina analizadora en la cual está incluído el DATA monitor y allí se compararán.
CASO 9. El DATA de la palabra clave está incluído en el programa principal; se transmite como argumento en la sentencia CALL que la invoca, a la rutina analiza-

dora en la cual está incluído el DATA monitor y allí se compararán.

CASO 10. El DATA de la palabra clave está incluído en el programa principal; se transmite como argumento en la sentencia CALL que la invoca, a la rutina analizadora en la cual se comparará con el DATA monitor. Este está incluído en un BLOCK DATA y se transmite a esta rutina por medio de una sentencia COMMON.

CASO 11. El DATA de la palabra clave y el DATA monitor están incluídos en el mismo BLOCK DATA. Ambos se transmiten, por medio de una sentencia COMMON, a la rutina analizadora en donde se compararán.

CASO 12. El DATA de la palabra clave está incluído en la rutina analizadora. El DATA monitor está incluído en un BLOCK DATA; se transmite a dicha rutina por medio de una sentencia COMMON y allí se compararán.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 y 7.

El dato literal es la palabra AJT1 que ingresa al programa por formato de lectura, a la variable CLAVE, REAL*4, (CASO1) o bien está definida en la sentencia DATA de nombre CLAVE, REAL*4, (CASO 7). Esta tira de caracteres se comparará mediante el DO 10, con cada una de las palabras del DATA monitor de nombre ARUTIN, REAL*4 y de dimensión 4. Cuando encuentre la igualdad ramificará hacia una sentencia GO TO computada (marca 11) que hace las veces de distribuidora y transferirá el control del programa a la rutina de cálculo que corresponda.

Si por error del programador el DO 10 se agota sin encontrar ninguna igualdad, la sentencia GO TO 8 transferirá el control del programa hacia la escritura de una leyenda adecuada y desde allí saltará al STOP a terminar la ejecución.

Toda vez que se realizare una comparación de tiras de caracteres en una sentencia IF, las variables y vectores involucrados en la comparación, deberán ser del mismo tipo. La variable CLAVE y el vector ARUTIN son ambos REAL*4 por convención implícita.

CAP.II,PROGR.4,CASO 1. Se transfiere el control del programa a una rutina. La palabra AJT1 ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE. El DATA monitor de nombre ARUTIN está incluído en ese mismo programa y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION ARUTIN(4)
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      READ(5,100) CLAVE
      READ(5,102) A,B
```

```
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 9
    2 CALL AJUST1(A,B,C)
      GO TO 9
    3 CALL METOD1(A,B,C)
      GO TO 9
    4 CALL PRUEB1(A,B,C)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 STOP
  100 FORMAT(A4)
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
/*
//GO.SYSIN DD *
AJT1
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 2. Se transfiere el control dél programa a una rutina. La palabra AJT1 ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE ANALIZ por una sentencia COMMON. El DATA monitor de nombre ARUTIN está incluído en dicha rutina y allí se

comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/CARAC/CLAVE
      READ(5,100) CLAVE
      READ(5,102) A,B
      CALL ANALIZ(A,B,C)
      WRITE(6,101) C
      STOP
  100 FORMAT(A4)
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      SUBROUTINE ANALIZ(A,B,C)
      DIMENSION ARUTIN(4)
      COMMON/CARAC/CLAVE
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I))·GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
      GO TO 12
    3 CALL METOD1(A,B,C)
      GO TO 12
    4 CALL PRUEB1(A,B,C)
      GO TO 12
    8 WRITE(6,103)
   12 RETURN
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
```

```
/*
//GO.SYSIN DD *
AJT1
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 3. Se transfiere el control del programa a una rutina. La palabra AJT1 ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE ANALIZ como argumento en la sentencia CALL que la invoca. El DATA monitor de nombre ARUTIN está incluído en dicha rutina y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) CLAVE
      READ(5,102) A,B
      CALL ANALIZ(CLAVE,A,B,C)
      WRITE(6,101) C
      STOP
  100 FORMAT(A4)
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      SUBROUTINE ANALIZ(CLAVE,A,B,C)
      DIMENSION ARUTIN(4)
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
      GO TO 12
    3 CALL METOD1(A,B,C)
      GO TO 12
    4 CALL PRUEB1(A,B,C)
      GO TO 12
    8 WRITE(6,103)
   12 RETURN
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
```

```
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
/*
//GO.SYSIN DD *
AJT1
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 4. Se transfiere el control del programa a una rutina. La palabra AJT1 ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE ANALIZ como argumento en la sentencia CALL que la invoca. El DATA monitor de nombre ARUTIN está incluído en un BLOCK DATA y se transmite, por medio de una sentencia COMMON, a dicha rutina donde se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) CLAVE
      READ(5,102) A,B
      CALL ANALIZ(CLAVE,A,B,C)
      WRITE(6,101) C
      STOP
  100 FORMAT(A4)
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/ARUTIN(4)
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      END
C
      SUBROUTINE ANALIZ(CLAVE,A,B,C)
      COMMON/CARAC/ARUTIN(4)
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO (1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
```

```
      GO TO 12
    3 CALL METOD1(A,B,C)
      GO TO 12
    4 CALL PRUEB1(A,B,C)
      GO TO 12
    8 WRITE(6,103)
   12 RETURN
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
/*
//GO.SYSIN DD *
AJT1
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 5. Se transfiere el control del programa a una rutina. La palabra AJT1 ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE ANALIZ por una sentencia COMMON. El DATA monitor de nombre ARUTIN está incluído en un BLOCK DATA y se transmite por la misma sentencia COMMON, a dicha rutina en donde se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/CARAC/CLAVE,ARUTIN(4)
      READ(5,100) CLAVE
      READ(5,102) A,B
      CALL ANALIZ(A,B,C)
      WRITE(6,101) C
      STOP
  100 FORMAT(A4)
```

```
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/CLAVE,ARUTIN(4)
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      END
C
      SUBROUTINE ANALIZ(A,B,C)
      COMMON/CARAC/CLAVE,ARUTIN(4)
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
      GO TO 12
    3 CALL METOD1(A,B,C)
      GO TO 12
    4 CALL PRUEB1(A,B,C)
      GO TO 12
    8 WRITE(6,103)
   12 RETURN
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
/*
//GO.SYSIN DD *
AJT1
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 6. Se transfiere el control del programa a una rutina. La palabra AJT1 ingresa a la SUBROUTINE ANALIZ por medio de un formato de lectura, a la variable CLAVE. El DATA monitor de nombre ARUTIN está incluído en un BLOCK DATA y se transmite, por una sentencia COMMON a dicha rutina en donde se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,102) A,B
      CALL ANALIZ(A,B,C)
      WRITE(6,101) C
      STOP
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/ARUTIN(4)
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      END
C
      SUBROUTINE ANALIZ(A,B,C)
      COMMON/CARAC/ARUTIN(4)
      READ(5,100) CLAVE
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
      GO TO 12
    3 CALL METOD1(A,B,C)
      GO TO 12
    4 CALL PRUEB1(A,B,C)
      GO TO 12
    8 WRITE(6,103)
   12 RETURN
  100 FORMAT(A4)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
```

```
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
/*
//GO.SYSIN DD *
AJT1
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 7. Se transfiere el control del programa a una rutina. La palabra AJT1 está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE. El DATA monitor de nombre ARUTIN está incluído en ese mismo programa y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1  EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION ARUTIN(4)
      DATA CLAVE/'AJT1'/
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      READ(5,102) A,B
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 9
    2 CALL AJUST1(A,B,C)
      GO TO 9
    3 CALL METOD1(A,B,C)
      GO TO 9
    4 CALL PRUEB1(A,B,C)
      GO TO 9
    8 WRITE(6,103)
      GO TO 12
    9 WRITE(6,101) C
   12 STOP
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
```

```
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 8. Se transfiere el control del programa a una rutina. La palabra AJT1 está definida por medio de una sentencia DATA de nombre CLAVE, incluída en un BLOCK DATA. Desde allí se transmite por una sentencia COMMON a la SUBROUTINE ANALIZ. El DATA monitor de nombre ARUTIN está incluído en dicha rutina y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,102) A,B
      CALL ANALIZ(A,B,C)
      WRITE(6,101) C
      STOP
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/CLAVE
      DATA CLAVE/'AJT1'/
      END
C
      SUBROUTINE ANALIZ(A,B,C)
      DIMENSION ARUTIN(4)
      COMMON/CARAC/CLAVE
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO (1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
      GO TO 12
```

```
     3 CALL METOD1(A,B,C)
       GO TO 12
     4 CALL PRUEB1(A,B,C)
       GO TO 12
     8 WRITE(6,103)
    12 RETURN
   103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
       END
C
       SUBROUTINE CALC11(A,B,C)
       C=A+B
       RETURN
       END
C
       SUBROUTINE AJUST1(A,B,C)
       C=A-B
       RETURN
       END
C
       SUBROUTINE METOD1(A,B,C)
       C=A*B
       RETURN
       END
C
       SUBROUTINE PRUEB1(A,B,C)
       C=A/B
       RETURN
       END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 9. Se transfiere el control del programa a una rutina. La palabra AJT1 está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE. Desde allí se transmite como argumento en la sentencia CALL que la invoca, a la SUBROUTINE ANALIZ. El DATA monitor de nombre ARUTIN está incluído en dicha rutina y allí se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA CLAVE/'AJT1'/
      READ(5,102) A,B
      CALL ANALIZ(CLAVE,A,B,C)
      WRITE(6,101) C
      STOP
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      SUBROUTINE ANALIZ(CLAVE,A,B,C)
      DIMENSION ARUTIN(4)
```

```
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
      GO TO 12
    3 CALL METOD1(A,B,C)
      GO TO 12
    4 CALL PRUEB1(A,B,C)
      GO TO 12
    8 WRITE(6,103)
   12 RETURN
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 10. Se transfiere el control del programa a una rutina. La palabra AJT1 está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE. Desde allí se transmite a la SUBROUTINE ANALIZ como argumento en la sentencia CALL que la invoca. El DATA monitor de nombre ARUTIN está incluído en un BLOCK DATA y se transmite, por medio de una sentencia COMMON, a dicha rutina donde se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
```

```
      DATA CLAVE/'AJT1'/
      READ(5,102) A,B
      CALL ANALIZ(CLAVE,A,B,C)
      WRITE(6,101) C
      STOP
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/ARUTIN(4)
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      END
C
      SUBROUTINE ANALIZ(CLAVE,A,B,C)
      COMMON/CARAC/ARUTIN(4)
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
      GO TO 12
    3 CALL METOD1(A,B,C)
      GO TO 12
    4 CALL PRUEB1(A,B,C)
      GO TO 12
    8 WRITE(6,103)
   12 RETURN
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
/*
//GO.SYSIN DD *
 23.25 12.35
./*
//
```

CAP.II,PROGR.4,CASO 11. Se transfiere el control del programa a una rutina. La palabra AJT1 está definida por medio de una sentencia DATA de nombre CLAVE incluída en un BLOCK DATA. El DATA monitor de nombre ARUTIN está incluído en ese mismo BLOCK DATA. Ambos se transmiten por la misma sentencia COMMON a la SUBROUTINE ANALIZ en la cual se compararán.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,102) A,B
      CALL ANALIZ(A,B,C)
      WRITE(6,101) C
      STOP
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/CLAVE,ARUTIN(4)
      DATA CLAVE/'AJT1'/
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      END
C
      SUBROUTINE ANALIZ(A,B,C)
      COMMON/CARAC/CLAVE,ARUTIN(4)
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
      GO TO 12
    3 CALL METOD1(A,B,C)
      GO TO 12
    4 CALL PRUEB1(A,B,C)
      GO TO 12
    8 WRITE(6,103)
   12 RETURN
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
```

```
      RETURN
      END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAP.II,PROGR.4,CASO 12. Se transfiere el control del programa a una rutina. La palabra AJT1 está definida por medio de una sentencia DATA de nombre CLAVE en la SUBROUTINE ANALIZ. El DATA monitor de nombre ARUTIN está incluído en un BLOCK DATA y se transmite, por una sentencia COMMON, a dicha rutina en donde se comparará con la palabra clave.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,102) A,B
      CALL ANALIZ(A,B,C)
      WRITE(6,101) C
      STOP
  101 FORMAT(1X,E14.6)
  102 FORMAT(2F6.2)
      END
C
      BLOCK DATA
      COMMON/CARAC/ARUTIN(4)
      DATA ARUTIN/'CL11','AJT1','MET1','PRU1'/
      END
C
      SUBROUTINE ANALIZ(A,B,C)
      COMMON/CARAC/ARUTIN(4)
      DATA CLAVE/'AJT1'/
      DO 10 I=1,4
      IF(CLAVE.EQ.ARUTIN(I)) GO TO 11
   10 CONTINUE
      GO TO 8
   11 GO TO(1,2,3,4),I
    1 CALL CALC11(A,B,C)
      GO TO 12
    2 CALL AJUST1(A,B,C)
      GO TO 12
    3 CALL METOD1(A,B,C)
      GO TO 12
    4 CALL PRUEB1(A,B,C)
      GO TO 12
    8 WRITE(6,103)
   12 RETURN
  103 FORMAT(1X,'LA PALABRA CLAVE NO FIGURA EN EL DATA MONITOR')
      END
C
      SUBROUTINE CALC11(A,B,C)
      C=A+B
      RETURN
```

```
      END
C
      SUBROUTINE AJUST1(A,B,C)
      C=A-B
      RETURN
      END
C
      SUBROUTINE METOD1(A,B,C)
      C=A*B
      RETURN
      END
C
      SUBROUTINE PRUEB1(A,B,C)
      C=A/B
      RETURN
      END
/*
//GO.SYSIN DD *
 23.25 12.35
/*
//
```

CAPITULO III

INTERRUPCION Y ALTERACION SELECTIVA DEL ORDEN DE PROCESAMIENTO DE JUEGOS DE DATOS SECUENCIALES

PROGRAMA 5.

OBJETO: Un programa tiene numerosos juegos de datos secuenciales precedido cada uno por una tarjeta con un título de 4 caracteres. Se desea no alterar el lote de datos en cada corrida, pero sí interrumpir la ejecución al finalizar cualquiera de los juegos. Se lo conseguirá remplazando la tarjeta con el título por una tarjeta en blanco que desempeña el papel de centinela, a la altura en que se espera la interrupción. El programa cuenta a su vez con una palabra clave toda de blancos para realizar la comparación y localizar la tarjeta centinela.

CASOS CONSIDERADOS

A) La palabra clave ingresa al programa principal por medio de un formato de lectura.

CASO 1. El programa no tiene rutinas a las cuales se transmita la palabra clave.
CASO 2. La palabra clave se transmite a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. La palabra clave se transmite a una rutina de cálculo como argumento en la sentencia CALL que la invoca.

B) La palabra clave está definida por medio de una sentencia DATA o en el programa principal o bien en una subrutina BLOCK DATA.

CASO 4. El programa no tiene rutinas a las cuales se transmita la palabra clave.
CASO 5. La palabra clave está incluída en un BLOCK DATA y se transmite a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. La palabra clave se transmite a una rutina de cálculo como argumento en la sentencia CALL que la invoca.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 Y 4.

El dato literal es la palabra bbbb (4 blancos) que ingresa al programa por formato de lectura a la variable CLAVE, REAL*4, (CASO 1) o bien está definida en la sentencia DATA de nombre CLAVE, REAL*4, (CASO 4). La tira de caracteres que precede a cada juego de datos consta de letras, números (inclusive todos ceros) o blancos e ingresa a la variable, REAL*4, de nombre TITULO. Cada juego está compuesto de tres tarjetas con cuatro datos cada una.

El bucle 3, comprendido entre la marca 3 y la sentencia GO TO 3, se encargará de leer para cada juego, el título y los datos numéricos. Por medio de una sentencia IF se prueba si TITULO es igual a CLAVE (4 blancos); si no es, ejecutará

los cálculos previstos y volverá a la marca 3 a leer el próximo juego. Si TITULO resultara igual a CLAVE o sea el programa encontró la tarjeta centinela en blanco, se terminará la ejecución. En los ejemplos hay 4 juegos de datos de los cuales se ejecutarán el primero y segundo solamente en los CASOS 1 y 4; el primero, segundo y tercero solamente en los CASOS 2 y 5 y por último solamente el primer juego en los CASOS 3 y 6. Si no hubiera tarjeta centinela la sentencia READ de la marca 3 leerá todos los juegos que encuentre y terminará el programa cuando llegue a la sentencia delimitadora /*.

Toda vez que se realizare una comparación de tiras de caracteres en una sentencia IF, las variables involucradas en la comparación deberán ser del mismo tipo. Las variables CLAVE y TITULO son ambas REAL*4 por convención implícita.

CAP.III,PROGR.5,CASO 1. De los 4 juegos de datos secuenciales se ejecutarán los juegos 1 y 2 solamente. La palabra bbbb (4 blancos) ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE. El programa no tiene rutinas a las cuales se transmita esta palabra.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION W(3),X(3),Y(3),Z(3)
      READ(5,100) CLAVE
    3 READ(5,102,END=2) TITULO,(W(I),X(I),Y(I),Z(I),I=1,3)
      IF(TITULO.EQ.CLAVE) CALL EXIT
      DO 1 N=1,3
      A=X(N)-Z(N)/(W(N)+Y(N))
      B=A*A
      WRITE(6,101) B
    1 CONTINUE
      GO TO 3
    2 STOP
  100 FORMAT(A4)
  101 FORMAT(1X,E14.6)
  102 FORMAT(A4/(4F7.2))
      END
/*
//GO.SYSIN DD *
  Tarjeta en blanco=palabra clave
 -25
  -8.54   0.23  -2.57  21.78
  64.21  64.21 -56.32 -84.03
  -2.23 -74.02  47.18 -45.32
  00
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
  Tarjeta en blanco=tarjeta centinela
   1.03   0.23  77.56   0.98
   6.23   5.25  -2.33   5.75
```

```
 213.86    8.23    7.03 -18.01
SUMA
   2.63 -46.84    0.36  18.71
   1.86    8.23    7.03  14.01
 108.01 -18.01   52.06   1.64
/*
//
```

CAP.III,PROGR.5,CASO 2. De los 4 juegos de datos secuenciales se ejecutarán los juegos 1,2 y 3 solamente. La palabra bbbb (4 blancos) ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE CALCUL por una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/BLANCO/CLAVE
      READ(5,100) CLAVE
      CALL CALCUL
      STOP
  100 FORMAT(A4)
      END
C
      SUBROUTINE CALCUL
      DIMENSION W(3),X(3),Y(3),Z(3)
      COMMON/BLANCO/CLAVE
    3 READ(5,102,END=2) TITULO,(W(I),X(I),Y(I),Z(I),I=1,3)
      IF(TITULO.EQ.CLAVE) CALL EXIT
      DO 1 N=1,3
      A=X(N)-Z(N)/(W(N)+Y(N))
      B=A*A
      WRITE(6,101) B
    1 CONTINUE
      GO TO 3
    2 RETURN
  101 FORMAT(1X,E14.6)
  102 FORMAT(A4/(4F7.2))
      END
/*
//GO.SYSIN DD *
  Tarjeta en blanco=palabra clave
 -25
  -8.54   0.23  -2.57  21.78
  64.21  64.21 -56.32 -84.03
  -2.23 -74.02  47.18 -45.32
  00
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
REST
   1.03   0.23  77.56   0.98
   6.23   5.25  -2.33   5.75
 213.86   8.23   7.03 -18.01
  Tarjeta en blanco=tarjeta centinela
   2.63 -46.84   0.36  18.71
```

```
   1.86    8.23    7.03   14.01
 108.01  -18.01   52.06    1.64
/*
//
```

CAP.III,PROGR.5,CASO 3. De los 4 juegos de datos secuenciales se ejecutará el juego 1 solamente. La palabra bbbb (4 blancos) ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) CLAVE
      CALL CALCUL(CLAVE)
      STOP
  100 FORMAT(A4)
      END
C
      SUBROUTINE CALCUL(CLAVE)
      DIMENSION W(3),X(3),Y(3),Z(3)
    3 READ(5,102,END=2) TITULO,(W(I),X(I),Y(I),Z(I),I=1,3)
      IF(TITULO.EQ.CLAVE) CALL EXIT
      DO 1 N=1,3
      A=X(N)-Z(N)/(W(N)+Y(N))
      B=A*A
      WRITE(6,101) B
    1 CONTINUE
      GO TO 3
    2 RETURN
  101 FORMAT(1X,E14.6)
  102 FORMAT(A4/(4F7.2))
      END
/*
//GO.SYSIN DD *
  Tarjeta en blanco=palabra clave
 -25
  -8.54    0.23   -2.57   21.78
  64.21   64.21  -56.32  -84.03
  -2.23  -74.02   47.18  -45.32
  Tarjeta en blanco=tarjeta centinela
  -9.23    0.02    7.18  -45.32
   6.33   19.21   81.03   86.02
   0.82  -16.43   44.36   25.03
REST
   1.03    0.23   77.56    0.98
   6.23    5.25   -2.33    5.75
 213.86    8.23    7.03  -18.01
SUMA
   2.63  -46.84    0.36   18.71
   1.86    8.23    7.03   14.01
 108.01  -18.01   52.06    1.64
/*
//
```

CAP.III,PROGR.5,CASO 4. De los 4 juegos de datos secuenciales se ejecutarán los juegos 1 y 2 solamente. La palabra bbbb (4 blancos) está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE. El programa no tiene rutinas a las cuales se transmita esta palabra.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION W(3),X(3),Y(3),Z(3)
      DATA CLAVE/'    '/
    3 READ(5,102,END=2) TITULO,(W(I),X(I),Y(I),Z(I),I=1,3)
      IF(TITULO.EQ.CLAVE) CALL EXIT
      DO 1 N=1,3
      A=X(N)-Z(N)/(W(N)+Y(N))
      B=A*A
      WRITE(6,101) B
    1 CONTINUE
      GO TO 3
    2 STOP
  101 FORMAT(1X,E14.6)
  102 FORMAT(A4/(4F7.2))
      END
/*
//GO.SYSIN DD *
 -25
  -8.54   0.23  -2.57  21.78
  64.21  64.21 -56.32 -84.03
  -2.23 -74.02  47.18 -45.32
  00
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
  Tarjeta en blanco=tarjeta centinela
   1.03   0.23  77.56   0.98
   6.23   5.25  -2.33   5.75
 213.86   8.23   7.03 -18.01
SUMA
   2.63 -46.84   0.36  18.71
   1.86   8.23   7.03  14.01
 108.01 -18.01  52.06   1.64
/*
//
```

CAP.III,PROGR.5,CASO 5. De los 4 juegos de datos secuenciales se ejecutarán los juegos 1, 2 y 3 solamente. La palabra bbbb (4 blancos) está definida por medio de una sentencia DATA de nombre CLAVE, incluída en un BLOCK DATA, la cual se transmite a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      CALL CALCUL
      STOP
```

```
      END
C
      BLOCK DATA
      COMMON/BLANCO/CLAVE
      DATA CLAVE/'    '/
      END
C
      SUBROUTINE CALCUL
      DIMENSION W(3),X(3),Y(3),Z(3)
      COMMON/BLANCO/CLAVE
    3 READ(5,102,END=2) TITULO,(W(I),X(I),Y(I),Z(I),I=1,3)
      IF(TITULO.EQ.CLAVE) CALL EXIT
      DO 1 N=1,3
      A=X(N)-Z(N)/(W(N)+Y(N))
      B=A*A
      WRITE(6,101) B
    1 CONTINUE
      GO TO 3
    2 RETURN
  101 FORMAT(1X,E14.6)
  102 FORMAT(A4/(4F7.2))
      END
/*
//GO.SYSIN DD *
 -25
  -8.54   0.23  -2.57  21.78
  64.21  64.21 -56.32 -84.03
  -2.23 -74.02  47.18 -45.32
  00
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
REST
   1.03   0.23  77.56   0.98
   6.23   5.25  -2.33   5.75
 213.86   8.23   7.03 -18.01
  Tarjeta en blanco=tarjeta centinela
   2.63 -46.84   0.36  18.71
   1.86   8.23   7.03  14.01
 108.01 -18.01  52.06   1.64
/*
//
```

CAP.III,PROGR.5,CASO 6. De los 4 juegos de datos secuenciales se ejecutará el juego 1 solamente. La palabra bbbb(4 blancos) está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE la cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA CLAVE/'    '/
      CALL CALCUL(CLAVE)
      STOP
```

```
      END
C
      SUBROUTINE CALCUL(CLAVE)
      DIMENSION W(3),X(3),Y(3),Z(3)
    3 READ(5,102,END=2) TITULO,(W(I),X(I),Y(I),Z(I),I=1,3)
      IF(TITULO.EQ.CLAVE) CALL EXIT
      DO 1 N=1,3
      A=X(N)-Z(N)/(W(N)+Y(N))
      B=A*A
      WRITE(6,101) B
    1 CONTINUE
      GO TO 3
    2 RETURN
  101 FORMAT(1X,E14.6)
  102 FORMAT(A4/(4F7.2))
      END
/*
//GO.SYSIN DD *
 -25
  -8.54   0.23  -2.57  21.78
  64.21  64.21 -56.32 -84,03
  -2.23 -74.02  47.18 -45.32
  Tarjeta en blanco=tarjeta centinela
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
REST
   1.03   0.23  77.56   0.98
   6.23   5.25  -2.33   5.75
 213.86   8.23   7.03 -18.01
SUMA
   2.63 -46.84   0.36  18.71
   1.86   8.23   7.03  14.01
 108.01 -18.01  52.06   1.64
/*
//
```

PROGRAMA 6.

OBJETO: El programa tiene tres juegos de datos secuenciales de los cuales siempre se ejecutará el primer juego. Intercalada entre el primero y el segundo lote hay una tarjeta centinela con una tira de caracteres. Si esta tira fuera igual a una palabra clave se ejecutarán los juegos segundo y tercero. Si no fuera igual el programa leerá sin ejecutar las tarjetas correspondientes al segundo juego y ejecutará el tercero.

CASOS CONSIDERADOS

A) La palabra clave ingresa al programa principal por medio de un formato de lectura.

CASO 1. El programa no tiene rutinas a las cuales se transmita la palabra clave.
CASO 2. La palabra clave se transmite a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. La palabra clave se transmite a una rutina de cálculo como argumento en la sentencia CALL que la invoca.

B) La palabra clave está definida por medio de una sentencia DATA en el programa principal o bien en una subrutina BLOCK DATA.

CASO 4. El programa no tiene rutinas a las cuales se transmita la palabra clave.
CASO 5. La palabra clave está incluída en un BLOCK DATA y se transmite a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. La palabra clave se transmite a una rutina de cálculo como argumento en la sentencia CALL que la invoca.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 Y 4.

La palabra clave es la tira de caracteres SIGA la cual puede ingresar al programa por formato de lectura, a la variable CLAVE, REAL*4, (CASO 1) o bien estar definida en una sentencia DATA de nombre CLAVE, REAL*4, (CASO 4). El programa procesará 3 juegos de datos secuenciales que constan de 5,3 y 1 tarjetas respectivamente.

Una vez ingresada la palabra clave por cualquiera de las dos maneras mencionadas, se leerán 5 tarjetas y se ejecutará el primer juego de datos. A continuación se leerá la tira de caracteres de la tarjeta centinela que ingresará a la variable, REAL*4, de nombre CENTIN. Se compararán en una sentencia IF las variables CLAVE y CENTIN: si son iguales (como se ejemplifica en los CASOS 1,3 y 6) se eje-

cutarán secuencialmente los juegos de datos segundo y tercero. Si la tarjeta centinela contiene cualquier otra palabra que no sea SIGA, como por ejemplo blancos en el CASO 2, JUMP en el CASO 4 o DAT3 en el CASO 5, como resultado de la comparación entre las variables CLAVE y CENTIN en la sentencia IF, el control del programa se transferirá a la marca 1. Allí la sentencia READ(5,102) con su formato 102 FORMAT(//) producirá el efecto de saltear 3 tarjetas o sea todo el juego de datos 2, que por consiguiente no se ejecutará. Por últiomo se leerá una tarjeta y se ejecutará el juego tercero de datos.

Toda vez que se realizare una comparación de tiras de caracteres en una sentencia IF, las variables involucradas en la comparación deberán ser del mismo tipo. Las variables CLAVE y CENTIN son ambas REAL*4 por convención implícita.

NOTA: El formato de lectura con barras permitirá saltear cualquier número de tarjetas en los datos que están a continuación de la sentencia de control //GO.SYSIN DD *.

Algunos ejemplos ilustrarán su uso:

```
      READ(5,200)
  200 FORMAT(//////////////)
```

salteará 1 tarjeta por cada barra más 1 tarjeta al encontrar el paréntesis izquierdo que sigue inmediatamente a la palabra FORMAT, o sea 14 en total.

```
      READ(5,200)
  200 FORMAT(99(/))
```

salteará 99 veces 1 tarjeta más 1 tarjeta al encontrar el paréntesis izquierdo que sigue inmediatamente a la palabra FORMAT, o se 100 en total.

```
      READ(5,200)
  200 FORMAT(50(//))
```

salteará 50 veces 2 tarjetas más 1 tarjeta al encontrar el paréntesis izquierdo que sigue inmediatamente a la palabra FORMAT, o sea 101 en total.

CAP.III,PROGR.6,CASO 1. Se ejecutan secuencialmente los tres juegos de datos porque la tira de caracteres en la tarjeta centinela es igual a la palabra clave SIGA. Esta palabra ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE. El programa no tiene rutinas a las cuales se transmita dicha palabra.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION W(5),X(5),Y(5),Z(5)
      READ(5,101) CLAVE
      READ(5,100) (W(I),X(I),Y(I),Z(I),I=1,5)
      VACIO=(X(3)+Z(5)-W(1))/(W(2)*Y(5))
```

```
      VACIO=VACIO*(X(1)+Y(3))
      READ(5,101) CENTIN
      IF(CLAVE.NE.CENTIN) GO TO 1
      READ(5,100) A1,B1,C1,D1
      READ(5,100) A2,B2,C2,D2
      READ(5,100) A3,B3,C3,D3
      VACIO=(VACIO/(A2+D3-C1))*(A1*C2+B3)
      GO TO 2
    1 READ(5,102)
    2 READ(5,100) X(3),Y(3),W(4),Z(4)
      VACIO=(VACIO+X(3)-Y(3))/(W(4)-Z(4))
      WRITE(6,103) VACIO
      STOP
  100 FORMAT(4F7.2)
  101 FORMAT(A4)
  102 FORMAT(//)
  103 FORMAT(1X,'EL RESULTADO ES =',E14.6)
      END
/*
//GO.SYSIN DD *
SIGA  (es la palabra clave leída)
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
  11.63 -28.45   6.32  78.07
 107.01 -18.01  52.06 -45.64
SIGA  (es la tarjeta centinela)
 -21.32  -2.36  -2.36 -23 89
 -10.32  -2.36  59.36  23.89
 107.01 -18.01  22.06 -45.64
 -10.32  -0.36  59.36  23.89
/*
//
```

CAP.III,PROGR.6,CASO 2. Se ejecutan los juegos de datos 1 y 3 solamente porque la tira de caracteres en la tarjeta centinela es una palabra de 4 blancos, distinta por consiguiente de la palabra clave SIGA. Esta palabra ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE CALCUL por una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/CARAC/CLAVE
      READ(5,101) CLAVE
      CALL CALCUL
      STOP
  101 FORMAT(A4)
      END
C
      SUBROUTINE CALCUL
      DIMENSION W(5),X(5),Y(5),Z(5)
      COMMON/CARAC/CLAVE
      READ(5,100) (W(I),X(I),Y(I),Z(I),I=1,5)
```

```
      VACIO=(X(3)+Z(5)-W(1))/(W(2)*Y(5))
      VACIO=VACIO*(X(1)+Y(3))
      READ(5,101) CENTIN
      IF(CLAVE.NE.CENTIN) GO TO 1
      READ(5,100) A1,B1,C1,D1
      READ(5,100) A2,B2,C2,D2
      READ(5,100) A3,B3,C3,D3
      VACIO=(VACIO/(A2+D3-C1))*(A1*C2+B3)
      GO TO 2
    1 READ(5,102)
    2 READ(5,100) X(3),Y(3),W(4),Z(4)
      VACIO=(VACIO+X(3)-Y(3))/(W(4)-Z(4))
      WRITE(6,103) VACIO
      RETURN
  100 FORMAT(4F7.2)
  101 FORMAT(A4)
  102 FORMAT(//)
  103 FORMAT(1X,'EL RESULTADO ES=',E14.6)
      END
/*
//GO.SYSIN DD *
SIGA  (es la palabra clavé leída)
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
  11.63 -28.45   6.32  78.07
 107.01 -18.01  52.06 -45.64
      (es la tarjeta centinela)
 -21.32  -2.36  -2.36 -23.89
 -10.32  -2.36  59.36  23.89
 107.01 -18.01  22.06 -45.64
 -10.32  -0.36  59.36  23.89
/*
//
```

CAP.III,PROGR.6,CASO 3. Se ejecutan secuencialmente los tres juegos de datos porque la tira de caracteres en la tarjeta centinela es igual a la palabra clave SIGA. Esta palabra ingresa al programa principal por medio de un formato de lectura, a la variable CLAVE la cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,101) CLAVE
      CALL CALCUL(CLAVE)
      STOP
  101 FORMAT(A4)
      END
C
      SUBROUTINE CALCUL(CLAVE)
      DIMENSION W(5),X(5),Y(5),Z(5)
      READ(5,100) (W(I),X(I),Y(I),Z(I),I=1,5)
      VACIO=(X(3)+Z(5)-W(1))/(W(2)*Y(5))
```

```
      VACIO=VACIO*(X(1)+Y(3))
      READ(5,101) CENTIN
      IF(CLAVE.NE.CENTIN) GO TO 1
      READ(5,100) A1,B1,C1,D1
      READ(5,100) A2,B2,C2,D2
      READ(5,100) A3,B3,C3,D3
      VACIO=(VACIO/(A2+D3-C1))*(A1*C2+B3)
      GO TO 2
    1 READ(5,102)
    2 READ(5,100) X(3),Y(3),W(4),Z(4)
      VACIO=(VACIO+X(3)-Y(3))/(W(4)-Z(4))
      WRITE(6,103) VACIO
      RETURN
  100 FORMAT(4F7.2)
  101 FORMAT(A4)
  102 FORMAT(//)
  103 FORMAT(1X,'EL RESULTADO ES=',E14.6)
      END
/*
//GO.SYSIN DD *
SIGA  (es la palabra clave leída)
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
  11.63 -28.45   6.32  78.07
 107.01 -18.01  52.06 -45.64
SIGA  (es la tarjeta centinela)
 -21.32  -2.36  -2.36 -23.89
 -10.32  -2.36  59.36  23.89
 107.01 -18.01  22.06 -45.64
 -10.32  -0.36  59.36  23.89
/*
//
```

CAP.III,PROGR.6,CASO 4. Se ejecutan los juegos de datos 1 y 3 solamente porque la tira de caracteres en la tarjeta centinela es la palabra JUMP, distinta por consiguiente de la palabra clave SIGA. Esta palabra está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE. El programa no tiene rutinas a las cuales se transmita dicha palabra.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION W(5),X(5),Y(5),Z(5)
      DATA CLAVE/'SIGA'/
      READ(5,100) (W(I),X(I),Y(I),Z(I),I=1,5)
      VACIO=(X(3)+Z(5)-W(1))/(W(2)*Y(5))
      VACIO=VACIO*(X(1)+Y(3))
      READ(5,101) CENTIN
      IF(CLAVE.NE.CENTIN) GO TO 1
      READ(5,100) A1,B1,C1,D1
      READ(5,100) A2,B2,C2,D2
      READ(5,100) A3,B3,C3,D3
      VACIO=(VACIO/(A2+D3-C1))*(A1*C2+B3)
      GO TO 2
```

```
    1 READ(5,102)
    2 READ(5,100) X(3),Y(3),W(4),Z(4)
      VACIO=(VACIO+X(3)-Y(3))/(W(4)-Z(4))
      WRITE(6,103) VACIO
      STOP
  100 FORMAT(4F7.2)
  101 FORMAT(A4)
  102 FORMAT(//)
  103 FORMAT(1X,'EL RESULTADO ES=',E14.6)
      END
/*
//GO.SYSIN DD *
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
  11.63 -28.45   6.32  78.07
 107.01 -18.01  52.06 -45.64
JUMP  (es la tarjeta centinela)
 -21.32  -2.36  -2.36 -23.89
 -10.32  -2.36  59.36  23.89
 107.01 -18.01  22.06 -45.64
 -10.32  -0.36  59.36  23.89
/*
//
```

CAP.III,PROGR.6,CASO 5. Se ejecutan los juegos de datos 1 y 3 solamente porque la tira de caracteres en la tarjeta centinela es la palabra DAT3, distinta por consiguiente de la palabra clave SIGA. Esta palabra está definida por medio de una sentencia DATA de nombre CLAVE, incluída en un BLOCK DATA y se transmite a la SUBROUTINE CALCUL por una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/CARAC/CLAVE
      CALL CALCUL
      STOP
      END
C
      BLOCK DATA
      COMMON/CARAC/CLAVE
      DATA CLAVE/'SIGA'/
      END
C
      SUBROUTINE CALCUL
      COMMON/CARAC/CLAVE
      DIMENSION W(5),X(5),Y(5),Z(5)
      READ(5,100) (W(I),X(I),Y(I),Z(I),I=1,5)
      VACIO=(X(3)+Z(5)-W(1))/(W(2)*Y(5))
      VACIO=VACIO*(X(1)+Y(3))
      READ(5,101) CENTIN
      IF(CLAVE.NE.CENTIN) GO TO 1
      READ(5,100) A1,B1,C1,D1
      READ(5,100) A2,B2,C2,D2
```

```
      READ(5,100) A3,B3,C3,D3
      VACIO=(VACIO/(A2+D3-C1))*(A1*C2+B3)
      GO TO 2
    1 READ(5,102)
    2 READ(5,100) X(3),Y(3),W(4),Z(4)
      VACIO=(VACIO+X(3)-Y(3))/(W(4)-Z(4))
      WRITE(6,103) VACIO
      RETURN
  100 FORMAT(4F7.2)
  101 FORMAT(A4)
  102 FORMAT(//)
  103 FORMAT(1X,'EL RESULTADO ES =',E14.6)
      END
/*
//GO.SYSIN DD *
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
  11.63 -28.45   6.32  78.07
 107.01 -18.01  52.06 -45.64
DAT3  (es la tarjeta centinela)
 -21.32  -2.36  -2.36 -23·89
 -10.32  -2.36  59.36  23.89
 107.01 -18.01  22.06 -45.64
 -10.32  -0.36  59.36  23.89
/*
//
```

CAP.III,PROGR.6,CASO 6. Se ejecutan secuencialmente los tres juegos de datos porque la tira de caracteres en la tarjeta centinela es igual a la palabra clave SIGA. Esta palabra está definida en el programa principal por medio de una sentencia DATA de nombre CLAVE, la cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA/CLAVE/'SIGA'/
      CALL CALCUL(CLAVE)
      STOP
      END
C
      SUBROUTINE CALCUL(CLAVE)
      DIMENSION W(5),X(5),Y(5),Z(5)
      READ(5,100) (W(I),X(I),Y(I),Z(I),I=1,5)
      VACIO=(X(3)+Z(5)-W(1))/(W(2)*Y(5))
      VACIO=VACIO*(X(1)+Y(3))
      READ(5,101) CENTIN
      IF(CLAVE.NE.CENTIN) GO TO 1
      READ(5,100) A1,B1,C1,D1
      READ(5,100) A2,B2,C2,D2
      READ(5,100) A3,B3,C3,D3
      VACIO=(VACIO/(A2+D3-C1))*(A1*C2+B3)
      GO TO 2
```

```
    1 READ(5,102)
    2 READ(5,100) X(3),Y(3),W(4),Z(4)
      VACIO=(VACIO+X(3)-Y(3))/(W(4)-Z(4))
      WRITE(6,103) VACIO
      RETURN
  100 FORMAT(4F7.2)
  101 FORMAT(A4)
  102 FORMAT(//)
  103 FORMAT(1X,'EL RESULTADO ES =',E14.6)
      END
/*
//GO.SYSIN DD *
  -9.23   0.02   7.18 -45.32
   6.33  19.21  81.03  86.02
   0.82 -16.43  44.36  25.03
  11.63 -28.45   6.32  78.07
 107.01 -18.01  52.06 -45.64
SIGA  (es la tarjeta centinela)
 -21.32  -2.36  -2.36 -23.89
 -10.32  -2.36  59.36  23.89
 107.01 -18.01  22.06 -45.64
 -10.32  -0.36  59.36  23.89
/*
//
```

PROGRAMA 7.

OBJETO: Un programa tiene numerosos juegos de datos secuenciales, encabezado cada uno de ellos por una tarjeta con un título de cuatro caracteres. Una tira de caracteres que desempeña el papel de selectora, contiene los títulos de los juegos a procesar. El programa ejecutará solamente los juegos de datos solicitados y salteará los demás.

CASOS CONSIDERADOS

A) La tira selectora de títulos ingresa al programa principal por medio de un formato de lectura.

CASO 1. El programa no tiene rutinas a las cuales se transmita la tira selectora de títulos.
CASO 2. La tira selectora de títulos se transmite a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. La tira selectora de títulos se transmite a una rutina de cálculo como argumento en la sentencia CALL que la invoca.

B) La tira selectora de títulos está definida por medio de una sentencia DATA o bien en el programa principal o en una subrutina BLOCK DATA.

CASO 4. El programa no tiene rutinas a las cuales se transmita la tira selectora de títulos.
CASO 5. La tira selectora de títulos está incluída en un BLOCK DATA y se transmite a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. La tira selectora de títulos se transmite a una rutina de cálculo como argumento en la sentencia CALL que la invoca.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 Y 4.

El programa tiene 6 juegos de datos cada uno de los cuales consta de una tarjeta con un título de 4 caractares y 3 tarjetas con números. Los títulos son: DAT1, DAT2,DAT3,DAT4,DAT5 y DAT6 los cuales ocuparán sucesivamente al leer cada juego, la variable, REAL*4, de nombre TITULO.

La tira selectora de títulos es un vector, REAL*4, de dimensión 3, que ingresa al programa por formato de lectura a la variable de nombre SELECT o bien está definida en una sentencia DATA, REAL*4, de nombre SELECT y dimensión 3.

El programa buscará el primer título salteándose, sin procesarlas, todas las tarjetas que lo preceden y cuando lo encuentre ejecutará el juego de datos co-

rrespondiente. A continuación buscará el título siguiente, salteando nuevamente sin procesar las tarjetas que no interesen y cuando lo encuentre, ejecutará ese juego. Así procederá hasta agotar los títulos indicados en la tira selectora, ejecutando DAT2, DAT5 y DAT6 en el CASO 1; DAT1, DAT2 y DAT4 en el CASO 4.

El bucle 7 comprendido entre la marca 7 y la sentencia GO TO 7 contiene el DO 1 y el DO 2. El primero de ellos se ocupará de buscar el título por comparación entre TITULO y SELECT(I), en donde I valdrá sucesivamente 1, 2 y 3. Si no lo encontrara salteará las 3 tarjetas de ese juego, que no procesará, por medio del par de sentencias

```
      READ(5,101)
  101 FORMAT(//)
```

e irá a leer el título siguiente. Si lo encontrara irá a la marca 3 a sumar una unidad al índice I y a continuación entrar en el DO 2.

Este se encargará de leer los datos numéricos y efectuar los cálculos previstos. El bucle 7 terminará cuando la sentencia READ(5,100,END=6) TITULO se encuentre con la sentencia delimitadora /* al final del lote.

Si dos títulos pedidos son contiguos el programa no saltea tarjetas. Se supone que los juegos de datos tienen todos la misma estructura, es decir, el mismo número de tarjetas. La sentencia GO TO 6 se coloca por precaución; en efecto, si por error del usuario el DO 1 se agotara sin haber encontrado el título buscado, el programa bifurcará hacia el STOP y terminará.

NOTA: El formato de lectura con barras permitirá saltear cualquier número de tarjetas en los datos que están a continuación de la sentencia de control //GO.SYSIN DD *.

Algunos ejemplos ilustrarán su uso:

```
      READ(5,200)
  200 FORMAT(//////////////)
```

salteará 1 tarjeta por cada barra más 1 tarjeta al encontrar el paréntesis izquierdo que sigue inmediatamente a la palabra FORMAT, o sea 14 en total.

```
      READ(5,200)
  200 FORMAT(99(/))
```

salteará 99 veces 1 tarjeta más 1 tarjeta al encontrar el paréntesis izquierdo que sigue inmediatamente a la palabra FORMAT, o sea 100 en total.

```
      READ(5,200)
  200 FORMAT(50(//))
```

salteará 50 veces 2 tarjetas más 1 tarjeta al encontrar el paréntesis izquierdo que sigue inmediatamente a la palabra FORMAT, o sea 101 en total.

CAP.III,PROGR.7,CASO 1. Se ejecutan los juegos de datos 2, 5 y 6. La tira selec-

tora de títulos DAT2DAT5DAT6 ingresa al programa principal por medio de un formato de lectura, al vector SELECT,REAL*4 y de dimensión 3. El programa no tiene rutinas a las cuales se transmita esta tira selectora.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION SELECT(3)
      READ(5,105) SELECT
      I=1
    7 DO 1 J=1,6
      READ(5,100,END=6) TITULO
      IF(TITULO.EQ.SELECT(I)) GO TO 3
      READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I=I+1
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 STOP
  100 FORMAT(A4)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
  105 FORMAT(3A4)
      END
/*
//GO.SYSIN DD *
DAT2DAT5DAT6
DAT1
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
DAT2
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
  0.9 -0.5  0.3  0.8 -0.5
DAT3
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -7.0  0.9 -8.2  7.3  3.0
DAT4
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.0 -0.5  5.8  0.6
DAT5
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
DAT6
  6.4  8.6  7.2  0.2  7.7
```

```
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.7,CASO 2. Se ejecutan los juegos de datos 1, 3 y 6.La tira selectora de títulos DAT1DAT3DAT6 ingresa al programa principal por medio de un formato de lectura, al vector SELECT, REAL*4 y de dimensión 3, el cual se transmite a la SUBROUTINE CALCUL por una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/CARAC/SELECT(3)
      READ(5,105) SELECT
      CALL CALCUL
      STOP
  105 FORMAT(3A4)
      END
C
      SUBROUTINE CALCUL
      COMMON/CARAC/SELECT(3)
      I=1
    7 DO 1 J=1,6
      READ(5,100,END=6) TITULO
      IF(TITULO.EQ.SELECT(I)) GO TO 3
      READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I=I+1
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 RETURN
  100 FORMAT(A4)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
      END
/*
//GO.SYSIN DD *
DAT1DAT3DAT6
DAT1
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
DAT2
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
```

```
  0.9 -0.5  0.3  0.8 -0.5
DAT3
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -7.0  0.9 -8.2  7.3  3.0
DAT4
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.0 -0.5  5.8  0.6
DAT5
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
DAT6
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.7,CASO 3. Se ejecutan los juegos de datos 1, 2 y 5. La tira selectora de títulos DAT1DAT2DAT5 ingresa al programa principal por medio de un formato de lectura, al vector SELECT,REAL*4 y de dimensión 3, el cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION SELECT(3)
      READ(5,105) SELECT
      CALL CALCUL(SELECT)
      STOP
  105 FORMAT(3A4)
      END
C
      SUBROUTINE CALCUL(SELECT)
      DIMENSION SELECT(3)
      I=1
    7 DO 1 J=1,6
      READ(5,100,END=6) TITULO
      IF(TITULO.EQ.SELECT(I)) GO TO 3
      READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I=I+1
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 RETURN
```

```
  100 FORMAT(A4)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
      END
/*
//GO.SYSIN DD *
DAT1DAT2DAT5
DAT1
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
DAT2
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
  0.9 -0.5  0.3  0.8 -0.5
DAT3
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -7.0  0.9 -8.2  7.3  3.0
DAT4
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.0 -0.5  5.8  0.6
DAT5
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
DAT6
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.7,CASO 4. Se ejecutan los juegos de datos 1, 2 y 4. La tira selectora de títulos DAT1DAT2DAT4 está definida en el programa principal por medio de una sentencia DATA, REAL*4, de nombre SELECT y dimensión 3. El programa no tiene rutinas a las cuales se transmita esta tira selectora.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION SELECT(3)
      DATA SELECT/'DAT1','DAT2','DAT4'/
      I=1
    7 DO 1 J=1,6
      READ(5,100,END=6) TITULO
      IF(TITULO.EQ.SELECT(I)) GO TO 3
      READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I=I+1
      DO 2 M=1,3
```

```
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 STOP
  100 FORMAT(A4)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
      END
/*
//GO.SYSIN DD *
DAT1
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
DAT2
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
  0.9 -0.5  0.3  0.8 -0.5
DAT3
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -7.0  0.9 -8.2  7.3  3.0
DAT4
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.0 -0.5  5.8  0.6
DAT5
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
DAT6
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.7,CASO 5. Se ejecutan los juegos de datos 1, 3 y 5. La tira selectora de títulos DAT1DAT3DAT5 está definida por medio de una sentencia DATA de nombre SELECT, REAL*4, de dimensión 3, incluída en un BLOCK DATA, la cual se transmite a la SUBROUTINE CALCUL por una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      CALL CALCUL
      STOP
      END
C
      BLOCK DATA
```

```
      COMMON/CARAC/SELECT(3)
      DATA SELECT/'DAT1','DAT3','DAT5'/
      END
C
      SUBROUTINE CALCUL
      COMMON/CARAC/SELECT(3)
      I=1
    7 DO 1 J=1,6
      READ(5,100,END=6) TITULO
      IF(TITULO.EQ.SELECT(I)) GO TO 3
      READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I=I+1
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 RETURN
  100 FORMAT(A4)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
      END
/*
//GO.SYSIN DD *
DAT1
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
DAT2
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
  0.9 -0.5  0.3  0.8 -0.5
DAT3
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -7.0  0.9 -8.2  7.3  3.0
DAT4
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.0 -0.5  5.8  0.6
DAT5
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
DAT6
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.7,CASO 6. Se ejecutan los juegos de datos 2, 4 y 6. La tira selectora de títulos DAT2DAT4DAT6 está definida en el programa principal por medio de una sentencia DATA de nombre SELECT, REAL*4, de dimensión 3, la cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION SELECT(3)
      DATA SELECT/'DAT2','DAT4','DAT6'/
      CALL CALCUL(SELECT)
      STOP
      END
C
      SUBROUTINE CALCUL(SELECT)
      DIMENSION SELECT(3)
      I=1
    7 DO 1 J=1,6
      READ(5,100,END=6) TITULO
      IF(TITULO.EQ.SELECT(I)) GO TO 3
      READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I=I+1
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 RETURN
  100 FORMAT(A4)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES=',F8.2)
      END
/*
//GO.SYSIN DD *
DAT1
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
DAT2
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
  0.9 -0.5  0.3  0.8 -0.5
DAT3
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -0.7  0.9 -8.2  7.3  3.0
DAT4
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
```

```
 -7.0 -8.0 -0.5  5.8  0.6
DAT5
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
DAT6
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

PROGRAMA 8.

OBJETO: Un programa tiene numerosos juegos de datos secuenciales, encabezado cada uno de ellos por una tarjeta con una leyenda de 14 caracteres. Una tira de caracteres que desempeña el papel de selectora, contiene las leyendas de los juegos a procesar. El programa ejecuta solamente los juegos solicitados y saltea los demás.

CASOS CONSIDERADOS

A) La tira selectora de leyendas ingresa al programa principal por medio de un formato de lectura.

CASO 1. El programa no tiene rutinas a las cuales se transmita la tira selectora de leyendas.
CASO 2. La tira selectora de leyendas se transmite a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. La tira selectora de leyendas se transmite a una rutina de cálculo como argumento en la sentencia CALL que la invoca.

B) La tira selectora de leyendas está definida por medio de una sentencia DATA o bien en el programa principal o en una subrutina BLOCK DATA.

CASO 4. El programa no tiene rutinas a las cuales se transmita la tira selectora de leyendas.
CASO 5. La tira selectora de leyendas está incluída en un BLOCK DATA y se transmite a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. La tira selectora de leyendas se transmite a una rutina de cálculo como argumento en la sentencia CALL que la invoca.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 Y 4.

El programa tiene 6 juegos de datos cada uno de los cuales consta de una tarjeta con una leyenda de 14 caracteres y 3 tarjetas con números. Las leyendas son las siguientes:

EX.1 H32 ARGON
EX.2 H32 NEON
EX.3 F25 ARGON
EX.4 R60 NEON
EX.5 J63 XENON
EX.6 J63 ARGON

las cuales ocuparán sucesivamente, al leer cada juego, el vector ALEYEN de dimensión 2 y declarado REAL*8.

La tira selectora de leyendas, que contiene tres de ellas, ingresa al programa por formato de lectura al vector SELECL, de dimensión 6, declarado REAL*8 o bien está definida en una sentencia DATA de nombre SELECL,REAL*8 y dimensión 6.

El programa buscará la primera leyenda salteándose, sin procesarlas, todas las tarjetas que la preceden y cuando la encuentre ejecutará el juego de datos correspondiente. A continuación buscará la leyenda siguiente salteando nuevamente sin procesar las tarjetas que no interesen y cuando la encuentre ejecutará ese juego. Así procederá hasta agotar las leyendas indicadas en la tira selectora EX.3 F25 ARGON EX.5 J63 XENON EX.6 J63 ARGON en el CASO 1, que solicita los juegos 3, 5 y 6 y en la tira EX.1 H32 ARGON EX.5 J63 XENON EX.6 J63 ARGON, en el CASO 4, que solicita los juegos 1, 5 y 6.

El bucle 7 comprendido entre la marca 7 y la sentencia GO TO 7 contiene el DO 1 y el DO 2. El primero de ellos se encargará de buscar la leyenda por comparación entre ALEYEN(1) y SELECL(I1) y entre ALEYEN(2) y SELECL(I2) como se explicará un poco más adelante. Si no la encontrara salteará las 3 tarjetas de ese juego, que no procesará, por medio del par de sentencias

```
      READ(5,101)
  101 FORMAT(//)
```

e irá a leer la leyenda siguiente. Si la encontrara irá a la marca 3 a sumar 2 unidades a los índices I1 e I2 y a continuación entrar en el DO 2. Este se encargará de leer los datos numéricos y efectuar los cálculos previstos.

El bucle 7 terminará cuando la sentencia READ(5,100,END=6) ALEYEN se encuentre con la sentencia delimitadora /* al final del lote de datos. El índice J valdrá hasta 6 para poder analizar las leyendas de los 6 juegos de datos.

El índice I1 tomará sucesivamente los valores 1, 3 y 5 y el índice I2 los valores 2, 4 y 6. De esta manera en la primera vuelta del bucle 7 cuando ALEYEN(1) se compare con SELECL(1) y ALEYEN(2) se compare con SELECL(2) en las sentencias IF que aparecen en el DO 1,se estará comparando la leyenda leída en el juego de datos con la primera leyenda indicada en la tira selectora. En la próxima vuelta del bucle 7 cuando ALEYEN(1) se compare con SELECL(3) y ALEYEN(2) se compare con SELECL(4), se estará comparando la leyenda leída con la segunda leyenda de la tira selectora y así sucesivamente. Nótese que si el primer IF no se cumpliera el programa se salteará el segundo ya que aún cuando este último sí se cumpliera, el vector ALEYEN ya no puede ser igual a la leyenda buscada.

Si dos leyendas pedidas son contiguas el programa no salteará tarjetas. Se supone que los juegos de datos tienen todos la misma estructura, es decir, el mismo número de tarjetas. La sentencia GO TO 6 se coloca por precaución; en efecto, si por error del usuario el DO 1 se agotara sin haber encontrado la leyenda buscada,

el programa bifurcará hacia el STOP y terminará.

NOTA: El formato de lectura con barras permitirá saltear cualquier número de tarjetas en los datos que están a continuación de la sentencia de control //GO.SYSIN DD *.

Algunos ejemplos ilustrarán su uso:

```
      READ(5,200)
  200 FORMAT(/////////////)
```

salteará 1 tarjeta por cada barra más 1 tarjeta al encontrar el paréntesis izquierdo que sigue inmediatamente a la palabra FORMAT, o sea 14 en total.

```
      READ(5,200)
  200 FORMAT(99(/))
```

salteará 99 veces 1 tarjeta más 1 tarjeta al encontrar el paréntesis izquierdo que sigue inmediatamente a la palabra FORMAT, o sea 100 en total.

```
      READ(5,200)
  200 FORMAT(50(//))
```

salteará 50 veces 2 tarjetas más 1 tarjeta al encontrar el paréntesis izquierdo que sigue inmediatamente a la palabra FORMAT, o sea 101 en total.

CAP.III,PROGR.8,CASO 1. Se ejecutan los juegos de datos 3,5 y 6. La tira selectora de leyendas EX.3 F25 ARGON EX.5 J63 XENON EX.6 J63 ARGON ingresa al programa principal por medio de un formato de lectura al vector SELECL, REAL*8 y dimensión 6. El programa no tiene rutinas a las cuales se transmita esta tira selectora.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 ALEYEN,SELECL
      DIMENSION ALEYEN(2),SELECL(6)
      READ(5,104) SELECL
      I1=1
      I2=2
    7 DO 1 J=1,6
      READ(5,100,END=6) ALEYEN
      IF(ALEYEN(1).NE.SELECL(I1)) GO TO 8
      IF(ALEYEN(2).EQ.SELECL(I2)) GO TO 3
    8 READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I1=I1+2
      I2=I2+2
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
```

```
     6 STOP
   100 FORMAT(2A8)
   101 FORMAT(//)
   102 FORMAT(5F5.1)
   103 FORMAT(1X,'LA SUMA ES =',F8.2)
   104 FORMAT(3(2A8))
       END
/*
//GO.SYSIN DD *
EX.3 F25 ARGON  EX.5 J63 XENON  EX.6 J63 ARGON  (es la tira selectora)
EX.1 H32 ARGON
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
EX.2 H32  NEON
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
  0.9 -0.5  0.3  0.8 -0.5
EX.3 F25 ARGON
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -0.7  0.9 -8.2  7.3  3.0
EX.4 R60  NEON
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.2 -0.5  5.8  0.6
EX.5 J63 XENON
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
EX.6 J63 ARGON
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.8,CASO 2. Se ejecutan los juegos de datos 1, 2 y 4. La tira selectora de leyendas EX.1 H32 ARGON EX.2 H32 NEON EX.4 R60 NEON ingresa al programa principal por medio de un formato de lectura, al vector SELECL, REAL*8, de dimensión 6, el cual se transmite a la SUBROUTINE CALCUL por una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 SELECL
      COMMON/CARAC/SELECL(6)
      READ(5,104) SELECL
      CALL CALCUL
      STOP
  104 FORMAT(3(2A8))
      END
C
```

```
      SUBROUTINE CALCUL
      REAL * 8 ALEYEN,SELECL
      DIMENSION ALEYEN(2)
      COMMON/CARAC/SELECL(6)
      I1=1
      I2=2
    7 DO 1 J=1,6
      READ(5,100,END=6) ALEYEN
      IF(ALEYEN(1).NE.SELECL(I1)) GO TO 8
      IF(ALEYEN(2).EQ.SELECL(I2)) GO TO 3
    8 READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I1=I1+2
      I2=I2+2
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 RETURN
  100 FORMAT(2A8)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
      END
/*
//GO.SYSIN DD *
EX.1 H32 ARGON  EX.2 H32  NEON  EX.4 R60  NEON  (es la tira selectora)
EX.1 H32 ARGON
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
EX.2 H32  NEON
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
  0.9 -0.5  0.3  0.8 -0.5
EX.3 F25 ARGON
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -0.7  0.9 -8.2  7.3  3.0
EX.4 R60  NEON
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.2 -0.5  5.8  0.6
EX.5 J63 XENON
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
EX.6 J63 ARGON
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.8,CASO 3. Se ejecutan los juegos de datos 1, 2 y 5. La tira selectora de leyendas EX.1 H32 ARGON EX.2 H32 NEON EX.5 J63 XENON ingresa al programa principal por medio de un formato de lectura, al vector SELECL, REAL*8, de dimensión 6, el cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 SELECL
      DIMENSION SELECL(6)
      READ(5,104) SELECL
      CALL CALCUL(SELECL)
      STOP
  104 FORMAT(3(2A8))
      END
C
      SUBROUTINE CALCUL(SELECL)
      REAL * 8 ALEYEN,SELECL
      DIMENSION ALEYEN(2),SELECL(6)
      I1=1
      I2=2
    7 DO 1 J=1,6
      READ(5,100,END=6) ALEYEN
      IF(ALEYEN(1).NE.SELECL(I1)) GO TO 8
      IF(ALEYEN(2).EQ.SELECL(I2)) GO TO 3
    8 READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I1=I1+2
      I2=I2+2
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 RETURN
  100 FORMAT( 2A8)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
      END
/*
//GO.SYSIN DD *
EX.1 H32 ARGON  EX.2 H32  NEON  EX.5 J63 XENON  (es la tira selectora)
EX.1 H32 ARGON
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
EX.2 H32  NEON
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
```

```
  0.9 -0.5  0.3  0.8 -0.5
EX.3 F25 ARGON
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -0.7  0.9 -8.2  7.3  3.0
EX.4 R60  NEON
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.2 -0.5  5.8  0.6
EX.5 J63 XENON
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
EX.6 J63 ARGON
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.8,CASO 4. Se ejecutan los juegos de datos 1, 5 y 6. La tira selectora de leyendas EX.1 H32 ARGON EX.5 J63 XENON EX.6 J63 ARGON está definida en el programa principal por medio de una sentencia DATA de nombre SELECL, REAL*8, de dimensión 6. El programa no tiene rutinas a las cuales se transmita esta tira selectora.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 ALEYEN,SELECL
      DIMENSION ALEYEN(2),SELECL(6)
      DATA SELECL/'EX.1 H32',' ARGON  ','EX.5 J63',' XENON  ','EX.6 J63'
     1  ,' ARGON  '/
      I1=1
      I2=2
    7 DO 1 J=1,6
      READ(5,100,END=6) ALEYEN
      IF(ALEYEN(1).NE.SELECL(I1)) GO TO 8
      IF(ALEYEN(2).EQ.SELECL(I2)) GO TO 3
    8 READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I1=I1+2
      I2=I2+2
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 STOP
```

```
  100 FORMAT(2A8)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
  104 FORMAT(3(2A8))
      END
/*
//GO.SYSIN DD *
EX.1 H32 ARGON
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
EX.2 H32  NEON
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
  0.9 -0.5  0.3  0.8 -0.5
EX.3 F25 ARGON
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -0.7  0.9 -8.2  7.3  3.0
EX.4 R60  NEON
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.2 -0.5  5.8  0.6
EX.5 J63 XENON
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
EX.6 J63 ARGON
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.8,CASO 5. Se ejecutan los juegos de datos 2, 4 y 6. La tira selectora de leyendas EX.2 H32 NEON EX.4 R60 NEON EX.6 J63 ARGON está definida por medio de una sentencia DATA de nombre SELECL, REAL*8, de dimensión 6, incluída en un BLOCK DATA y se transmite a la SUBROUTINE CALCUL por una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      CALL CALCUL
      STOP
      END
C
      BLOCK DATA
      REAL * 8 SELECL
      COMMON/CARAC/SELECL(6)
      DATA SELECL/'EX.2 H32',' NEON  ','EX.4 R60',' NEON  ','EX.6 J63'
     1  ,' ARGON  '/
      END
C
```

```
      SUBROUTINE CALCUL
      REAL * 8 ALEYEN,SELECL
      DIMENSION ALEYEN(2)
      COMMON/CARAC/SELECL(6)
      I1=1
      I2=2
    7 DO 1 J=1,6
      READ(5,100,END=6) ALEYEN
      IF(ALEYEN(1).NE.SELECL(I1)) GO TO 8
      IF(ALEYEN(2).EQ.SELECL(I2)) GO TO 3
    8 READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I1=I1+2
      I2=I2+2
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 RETURN
  100 FORMAT(2A8)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
  104 FORMAT(3,(2A8))
      END
/*
//GO.SYSIN DD *
EX.1 H32 ARGON
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
EX.2 H32  NEON
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
  0.9 -0.5  0.3  0.8 -0.5
EX.3 F25 ARGON
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -0.7  0.9 -8.2  7.3  3.0
EX.4 R60  NEON
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.2 -0.5  5.8  0.6
EX.5 J63 XENON
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
EX.6 J63 ARGON
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAP.III,PROGR.8,CASO 6. Se ejecutan los juegos de datos 2, 3 y 5. La tira selectora de leyendas EX.2 H32 NEON EX.3 F25 ARGON EX.5 J63 XENON está definida en el programa principal por medio de una sentencia DATA de nombre SELECL, REAL*8, de dimensión 6, la cual se transmite a la SUBROUTINE CALCUL como argumento en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 SELECL
      DIMENSION SELECL(6)
      DATA SELECL/'EX.2 H32',' NEON  ','EX.3 F25',' ARGON  ','EX.5 J63'
     1  ,' XENON  '/
      CALL CALCUL(SELECL)
      STOP
      END
C
      SUBROUTINE CALCUL(SELECL)
      REAL * 8 ALEYEN,SELECL
      DIMENSION ALEYEN(2);SELECL(6)
      I1=1
      I2=2
    7 DO 1 J=1,6
      READ(5,100,END=6) ALEYEN
      IF(ALEYEN(1).NE.SELECL(I1)) GO TO 8
      IF(ALEYEN(2).EQ.SELECL(I2)) GO TO 3
    8 READ(5,101)
    1 CONTINUE
      GO TO 6
    3 I1=I1+2
      I2=I2+2
      DO 2 M=1,3
      READ(5,102) A,B,C,D,E
      SUM1=A+B
      SUM2=C-D
      SUMA=SUM1*SUM2-E
      WRITE(6,103) SUMA
    2 CONTINUE
      GO TO 7
    6 RETURN
  100 FORMAT(2A8)
  101 FORMAT(//)
  102 FORMAT(5F5.1)
  103 FORMAT(1X,'LA SUMA ES =',F8.2)
  104 FORMAT(3(2A8))
      END
/*
//GO.SYSIN DD *
EX.1 H32 ARGON
  9.9  8.1  0.3  0.7 -0.7
  4.0  4.3  3.5  0.2 -0.7
  6.4  8.6  7.2  0.2  7.7
EX.2 H32  NEON
  2.8  5.7  9.8 -4.4 -9.9
 -3.7 -0.5  6.3  5.7 -7.3
```

```
  0.9 -0.5  0.3  0.8 -0.5
EX.3 F25 ARGON
  0.8  0.6  0.3  0.3 -0.5
  0.7  0.6  0.1  0.2  0.5
 -0.7  0.9 -8.2  7.3  3.0
EX.4 R60  NEON
  7.7  3.0  0.8  3.1 -5.4
  9.8  7.0  0.7 -6.6 -7.4
 -7.0 -8.2 -0.5  5.8  0.6
EX.5 J63 XENON
  5.5  1.3  5.4 -2.5  0.5
 -2.7 -3.8 23.0 24.6 -3.5
 -6.5 13.5 -2.9 20.4 62.4
EX.6 J63 ARGON
  6.4  8.6  7.2  0.2  7.7
 -3.7 -0.5  6.3  5.7 -7.3
 -7.0  0.9 -8.2  7.3  3.0
/*
//
```

CAPITULO IV

USO DE FORMATOS DE LECTURA QUE SE ELIGEN DURANTE LA EJECUCION DEL PROGRAMA Y OTROS EJEMPLOS DE FORMATOS DE LECTURA NO CONVENCIONALES

PROGRAMA 9.

OBJETO: Las tarjetas que constituyen los juegos de datos serán leídas algunas con formatos convencionales y otras con formatos que se elegirán durante la ejecución del programa, entre varios formatos disponibles, según los cálculos numéricos efectuados y la consiguiente decisión de sentencias IF de comparación de los resultados. Cuando las tiras de caracteres estén definidas en sentencias DATA, se usarán apóstrofos para encerrar dichas tiras.

CASOS CONSIDERADOS

A) Los formatos de lectura disponibles ingresan al programa principal por tarjeta leída.

CASO 1. El programa no tiene rutinas a las cuales se transmitan los formatos de lectura disponibles.
CASO 2. Los formatos de lectura disponibles se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. Los formatos de lectura disponibles se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

B) Los formatos de lectura disponibles están definidos por medio de una sentencia DATA o bien en el programa principal o en una subrutina BLOCK DATA. Las tiras de caracteres se encierran entre apóstrofos.

CASO 4. El programa no tiene rutinas a las cuales se transmitan los formatos de lectura disponibles.
CASO 5. Los formatos de lectura disponibles están incluídos en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. Los formatos de lectura disponibles se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 Y 4.

El programa tiene 3 juegos de datos numéricos que constan de 2 tarjetas cada uno: la primera es distinta en cada juego y se leerá con el formato convencional 101, la segunda tarjeta es igual en los 3 juegos pero leída con un formato diferente cada vez, que se elegirá durante la ejecución según ciertas condiciones numéricas y conducirá a distintos valores de las variables B y D.

El formato variable FORM es un vector vacío de dimensión 2, declarado REAL*8, que se llenará con los formatos disponibles FORM1,FORM2 o FORM3 según el caso. Es-

tos formatos también son vectores de dimensión 2 y declarados REAL*8 que contienen las tiras de caracteres

```
(7X,F7.2,F7.2)
(F7.2,7X,F7.2)
(F7.2,F7.2,7X)
```

e ingresarán al programa por tarjeta leída con el formato convencional 102 en el CASO 1 o bien estarán definidas en sentencias DATA como en el CASO 4.

El formato variable FORM funciona de la siguiente manera: si se hiciera FORM(1)=FORM1(1) y FORM(2)=FORM1(2) (ver el DO 8) la sentencia

```
READ(5,FORM) B,D
```

leerá la segunda tarjeta del juego de datos numéricos con el formato (7X,F7.2,F7.2); si se hiciera FORM(1)=FORM2(1) y FORM(2)=FORM2(2) (ver el DO 9) la sentencia READ ya mencionada leerá la segunda tarjeta con el formato (F7.2,7X,F7.2) y finalmente, si se hiciera FORM(1)=FORM3(1) y FORM(2)=FORM3(2) (ver el DO 10), la sentencia READ ya mencionada leerá la segunda tarjeta del juego de datos con el formato (F7.2,F7.2,7X).

Al principio del programa, en el DO 12 se leerán las variables NUMERO, B, C y D de las cuales las tres últimas serán utilizadas, en un primer cálculo, en las marcas 1, 2 o 3 según el valor del índice NUMERO. Estas tres ramas convergirán hacia la marca 4 en donde un grupo de tres sentencias IF pasarán el control del programa a las marcas 5, 6 o 7 en donde se preparará el formato de lectura que se va a utilizar para leer la segunda tarjeta de datos numéricos.

Con el primer juego de datos numéricos

```
 0  21.18  23.32  13.50
  6.24 -14.40  -4.74
```

el programa irá a la marca 2, después irá a la marca 7 y con el formato FORM(I)=FORM3(I) hará B=6.24 y D=-14.40 mientras C conservará su valor 23.32.

Con el segundo juego de datos numéricos

```
-7  19.31 -17.25  -3.85
  6.24 -14.40  -4.74
```

el programa irá a la marca 1, después irá a la marca 5 y con el formato FORM(I)=FORM1(I) hará B=-14.40 y D=-4.74 mientras C conservará su valor -17.25.

Con el tercer juego de datos numéricos

```
 7  12.29  15.48   8.76
  6.24 -14.40  -4.74
```

el programa irá a la marca 3, después irá a la marca 6 y con el formato FORM(I)=FORM2(I) hará B=6.24 y D=-4.74 mientras C conservará su valor 15.48.

El programa convergirá hacia la marca 11 en donde efectuará un último cálculo con los nuevos valores de la variables B y D y terminará. La sentencia GO TO 15 se coloca por precaución y pasará el control del programa al STOP en el caso en

que no se cumpliera ninguna de las condiciones exigidas por el grupo de 3 sentencias IF de la marca 4.

CAP.IV,PROGR.9,CASO 1. Los formatos de lectura disponibles

(7X,F7.2,F7.2)
(F7.2,7X,F7.2)
(F7.2,F7.2,7X)

ingresan al programa principal por tarjeta leída con un formato convencional, a los vectores FORM1, FORM2 y FORM3, todos REAL*8 y de dimensión 2. Algunas tarjetas de datos se leen con el formato variable FORM, el cual se hará igual a una de estas tres posibilidades. El programa no tiene rutinas a las cuales se transmitan los formatos de lectura disponibles.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM,FORM1,FORM2,FORM3
      DIMENSION FORM(2),FORM1(2),FORM2(2),FORM3(2)
      READ(5,102) FORM1
      READ(5,102) FORM2
      READ(5,102) FORM3
      DO 12 J=1,3
      READ(5,101) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 A=B+C
      GO TO 4
    2 A=B-C
      GO TO 4
    3 A=C-D
    4 IF(B.GT.C.AND.A.LE.3.0) GO TO 5
      IF(A.GT.3.0) GO TO 6
      IF(B.LE.C.AND.C.NE.D) GO TO 7
      WRITE(6,104)
      GO TO 15
    5 DO 8 I=1,2
    8 FORM(I)=FORM1(I)
      GO TO 11
    6 DO 9 I=1,2
    9 FORM(I)=FORM2(I)
      GO TO 11
    7 DO 10 I=1,2
   10 FORM(I)=FORM3(I)
   11 READ(5,FORM) B,D
      A=(A+B)/(C*D)
      WRITE(6,103) A
   12 CONTINUE
   15 STOP
  101 FORMAT(I3,3F7.2)
  102 FORMAT(2A8)
  103 FORMAT(1X,'A=',E14.6)
  104 FORMAT(1X,' NO CUMPLE NINGUNA DE LAS CONDICIONES DE LAS SENTENCIAS
     1 IF DE LA MARCA 4')
```

```
      END
/*
//GO.SYSIN DD *
 (7X,F7.2,F7.2)
 (F7.2,7X,F7.2)
 (F7.2,F7.2,7X)
  0  21.18  23.32  13.50
   6.24 -14.40  -4.74
 -7  19.31 -17.25  -3.85
   6.24 -14.40  -4.74
  7  12.29  15.48   8.76
   6.24 -14.40  -4.74
/*
//
```

CAP.IV,PROGR.9,CASO 2. Los formatos de lectura disponibles

```
(7X,F7.2,F7.2)
(F7.2,7X,F7.2)
(F7.2.F7.2,7X)
```

ingresan al programa principal por tarjeta leída con un formato convencional, a los vectores FORM1, FORM2 y FORM3, todos REAL*8 y de dimensión 2. Algunas tarjetas de datos se leen con el formato variable FORM, el cual se hará igual a una de estas tres posibilidades. Los formatos de lectura disponibles se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3
      COMMON/FORLEC/ FORM1(2),FORM2(2),FORM3(2)
      READ(5,102) FORM1
      READ(5,102) FORM2
      READ(5,102) FORM3
      CALL CALCUL
      STOP
  102 FORMAT(2A8)
      END
C
      SUBROUTINE CALCUL
      REAL * 8 FORM,FORM1,FORM2,FORM3
      DIMENSION FORM (2)
      COMMON/FORLEC/FORM1(2),FORM2(2),FORM3(2)
      DO 12 J=1,3
      READ(5,101) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 A=B+C
      GO TO 4
    2 A=B-C
      GO TO 4
    3 A=C-D
    4 IF(B.GT.C.AND.A.LE.3.0) GO TO 5
      IF(A.GT.3.0) GO TO 6
      IF(B.LE.C.AND.C.NE.D) GO TO 7
```

```
      WRITE(6,104)
      GO TO 15
    5 DO 8 I=1,2
    8 FORM(I)=FORM1(I)
      GO TO 11
    6 DO 9 I=1,2
    9 FORM(I)=FORM2(I)
      GO TO 11
    7 DO 10 I=1,2
   10 FORM(I)=FORM3(I)
   11 READ(5,FORM) B,D
      A=(A+B)/(C*D)
      WRITE(6,103) A
   12 CONTINUE
   15 RETURN
  101 FORMAT(I3,3F7.2)
  103 FORMAT(1X,'A=',E14.6)
  104 FORMAT(1X,' NO CUMPLE NINGUNA DE LAS CONDICIONES DE LAS SENTENCIAS
     1 IF DE LA MARCA 4')
      END
/*
//GO.SYSIN DD *
 (7X,F7.2,F7.2)
 (F7.2,7X,F7.2)
 (F7.2,F7.2,7X)
  0  21.18  23.32  13.50
   6.24 -14.40  -4.74
 -7  19.31 -17.25  -3.85
   6.24 -14.40  -4.74
  7  12.29  15.48   8.76
   6.24 -14.40  -4.74
/*
//
```

CAP.IV,PROGR.9,CASO 3. Los formatos de lectura disponibles

```
(7X,F7.2,F7.2)
(F7.2,7X,F7.2)
(F7.2,F7.2,7X)
```

ingresan al programa principal por tarjeta leída con un formato convencional, a los vectores FORM1, FORM2 y FORM3, todos REAL*8 y de dimensión 2. Algunas tarjetas de datos se leen con el formato variable FORM, el cual se hará igual a una de estas tres posibilidades. Los formatos de lectura disponibles se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3
      DIMENSION FORM1(2),FORM2(2),FORM3(2)
      READ(5,102) FORM1
      READ(5,102) FORM2
      READ(5,102) FORM3
```

```
      CALL CALCUL(FORM1,FORM2,FORM3)
      STOP
  102 FORMAT(2A8)
      END
C
      SUBROUTINE CALCUL(FORM1,FORM2,FORM3)
      REAL * 8 FORM,FORM1,FORM2,FORM3
      DIMENSION FORM(2),FORM1(2),FORM2(2),FORM3(2)
      DO 12 J=1,3
      READ(5,101) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 A=B+C
      GO TO 4
    2 A=B-C
      GO TO 4
    3 A=C-D
    4 IF(B.GT.C.AND.A.LE.3.0) GO TO 5
      IF(A.GT.3.0) GO TO 6
      IF(B.LE.C.AND.C.NE.D) GO TO 7
      WRITE(6,104)
      GO TO 15
    5 DO 8 I=1,2
    8 FORM(I)=FORM1(I)
      GO TO 11
    6 DO 9 I=1,2
    9 FORM(I)=FORM2(I)
      GO TO 11
    7 DO 10 I=1,2
   10 FORM(I)=FORM3(I)
   11 READ(5,FORM) B,D
      A=(A+B)/(C*D)
      WRITE(6,103) A
   12 CONTINUE
   15 RETURN
  101 FORMAT(I3,3F7.2)
  103 FORMAT(1X,'A=',E14.6)
  104 FORMAT(1X,' NO CUMPLE NINGUNA DE LAS CONDICIONES DE LAS SENTENCIAS
     1 IF DE LA MARCA 4')
      END
/*
//GO.SYSIN DD *
 (7X,F7.2,F7.2)
 (F7.2,7X,F7.2)
 (F7.2,F7.2,7X)
  0  21.18  23.32  13.50
   6.24 -14.40  -4.74
 -7  19.31 -17.25  -3.85
   6.24 -14.40  -4.74
  7  12.29  15.48   8.76
   6.24 -14.40  -4.74
/*
//
```

CAP.IV,PROGR.9,CASO 4. Los formatos de lectura disponibles

(7X,F7.2.F7.2)

(F7.2,7X,F7.2)
(F7.2,F7.2,7X)
están definidos en el programa principal por medio de sentencias DATA de nombres FORM1, FORM2 y FORM3, todas REAL*8 y de dimensión 2. Algunas tarjetas de datos se leen con el formato variable FORM, el cual se hará igual a una de estas tres posibilidades. El programa no tiene rutinas a las cuales se transmitan los formatos de lectura disponibles.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM,FORM1,FORM2,FORM3
      DIMENSION FORM(2),FORM1(2),FORM2(2),FORM3(2)
      DATA FORM1/'(7X,F7.2',',F7.2)'/
      DATA FORM2/'(F7.2,7X',',F7.2)'/
      DATA FORM3/'(F7.2,F7',',2,7X)'/
      DO 12 J=1,3
      READ(5,101) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 A=B+C
      GO TO 4
    2 A=B-C
      GO TO 4
    3 A=C-D
    4 IF(B.GT.C.AND.A.LE.3.0) GO TO 5
      IF(A.GT.3.0) GO TO 6
      IF(B.LE.C.AND.C.NE.D) GO TO 7
      WRITE(6,102)
      GO TO 15
    5 DO 8 I=1,2
    8 FORM(I)=FORM1(I)
      GO TO 11
    6 DO 9 I=1,2
    9 FORM(I)=FORM2(I)
      GO TO 11
    7 DO 10 I=1,2
   10 FORM(I)=FORM3(I)
   11 READ(5,FORM) B,D
      A=(A+B)/(C*D)
      WRITE(6,103) A
   12 CONTINUE
   15 RETURN
  101 FORMAT(I3,3F7.2)
  102 FORMAT(1X,'NO CUMPLE NINGUNA DE LAS CONDICIONES DE LAS SENTENCIAS
     1 IF DE LA MARCA 4')
  103 FORMAT(1X,'A=',E14.6)
      END
/*
//GO.SYSIN DD *
  0  21.18  23.32  13.50
   6.24 -14.40  -4.74
 -7  19.31 -17.25  -3.85
   6.24 -14.40  -4.74
  7  12.29  15.48   8.76
   6.24 -14.40  -4.74
/*
//
```

CAP.IV,PROGR.9,CASO 5. Los formatos de lectura disponibles

(7X,F7.2,F7.2)
(F7.2,7X,F7.2)
(F7.2,F7.2,7X)

están definidos por medio de sentencias DATA de nombres FORM1, FORM2 y FORM3, todas REAL*8 y de dimensión 2, incluídas en un BLOCK DATA. Algunas tarjetas de datos se leen con el formato variable FORM el cual se hará igual a una de estas 3 posibilidades. Los formatos de lectura disponibles se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      CALL CALCUL
      STOP
      END
C
      BLOCK DATA
      REAL * 8 FORM1,FORM2,FORM3
      COMMON/FORLEC/FORM1(2),FORM2(2),FORM3(2)
      DATA FORM1/'(7X,F7.2',',F7.2)'/
      DATA FORM2/'(F7.2,7X',',F7.2)'/
      DATA FORM3/'(F7.2,F7',',2,7X)'/
      END
C
      SUBROUTINE CALCUL
      REAL * 8 FORM,FORM1,FORM2,FORM3
      DIMENSION FORM(2)
      COMMON/FORLEC/FORM1(2),FORM2(2),FORM3(2)
      DO 12 J=1,3
      READ(5,101) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 A=B+C
      GO TO 4
    2 A=B-C
      GO TO 4
    3 A=C-D
    4 IF(B.GT.C.AND.A.LE.3.0) GO TO 5
      IF(A.GT.3.0) GO TO 6
      IF(B.LE.C.AND.C.NE.D) GO TO 7
      WRITE(6,102)
      GO TO 15
    5 DO 8 I=1,2
    8 FORM(I)=FORM1(I)
      GO TO 11
    6 DO 9 I=1,2
    9 FORM(I)=FORM2(I)
      GO TO 11
    7 DO 10 I=1,2
   10 FORM(I)=FORM3(I)
   11 READ(5,FORM) B,D
      A=(A+B)/(C*D)
      WRITE(6,103) A
   12 CONTINUE
   15 RETURN
```

```
  101 FORMAT(I3,3F7.2)
  102 FORMAT(1X,' NO CUMPLE NINGUNA DE LAS CONDICIONES DE LAS SENTENCIAS
     1 IF DE LA MARCA 4')
  103 FORMAT(1X,'A=',E14.6)
      END
/*
//GO.SYSIN DD *
  0  21.18  23.32  13.50
   6.24 -14.40  -4.74
 -7  19.31 -17.25  -3.85
   6.24 -14.40  -4.74
  7  12.29  15.48   8.76
   6.24 -14.40  -4.74
/*
//
```

CAP.IV,PROGR.9,CASO 6. Los formatos de lectura disponibles

```
(7X,F7.2,F7.2)
(F7.2,7X,F7.2)
(F7.2,F7.2,7X)
```

están definidos en el programa principal por medio de sentencias DATA de nombres FORM1, FORM2 y FORM3, todas REAL*8 y de dimensión 2. Algunas tarjetas de datos se leen con el formato variable FORM, el cual se hará igual a una de estas 3 posibilidades. Los formatos de lectura disponibles se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3
      DIMENSION FORM1(2),FORM2(2),FORM3(2)
      DATA FORM1/'(7X,F7.2',',F7.2)'/
      DATA FORM2/'(F7.2,7X',',F7.2)'/
      DATA FORM3/'(F7.2,F7',',2,7X)'/
      CALL CALCUL(FORM1,FORM2,FORM3)
      STOP
      END
C
      SUBROUTINE CALCUL(FORM1,FORM2,FORM3)
      REAL * 8 FORM,FORM1,FORM2,FORM3
      DIMENSION FORM(2),FORM1(2),FORM2(2),FORM3(2)
      DO 12 J=1,3
      READ(5,101) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 A=B+C
      GO TO 4
    2 A=B-C
      GO TO 4
    3 A=C-D
    4 IF(B.GT.C.AND.A.LE.3.0) GO TO 5
      IF(A.GT.3.0) GO TO 6
      IF(B.LE.C.AND.C.NE.D) GO TO 7
      WRITE(6,102)
      GO TO 15
```

```
    5 DO 8 I=1,2
    8 FORM(I)=FORM1(I)
      GO TO 11
    6 DO 9 I=1,2
    9 FORM(I)=FORM2(I)
      GO TO 11
    7 DO 10 I=1,2
   10 FORM(I)=FORM3(I)
   11 READ(5,FORM) B,D
      A=(A+B)/(C*D)
      WRITE(6,103) A
   12 CONTINUE
   15 RETURN
  101 FORMAT(I3,3F7.2)
  103 FORMAT(1X,'A=',E14.6)
  102 FORMAT(1X,' NO CUMPLE NINGUNA DE LAS CONDICIONES DE LAS SENTENCIAS
     1 IF DE LA MARCA 4')
      END
/*
//GO.SYSIN DD *
  0  21.18  23.32  13.50
   6.24 -14.40  -4.74
 -7  19.31 -17.25  -3.85
   6.24 -14.40  -4.74
  7  12.29  15 48   8.76
   6.24 -14 40  -4.74
/*
//
```

PROGRAMA 10.

OBJETO: Tres formatos de lectura no convencionales están específicamente destinados a leer literales, números enteros y números flotantes respectivamente, a lo largo de todo el programa. Cuando los formatos de lectura están definidos en el programa en sentencias DATA, se usa el código de conversión H, también llamado código HOLLERITH.

CASOS CONSIDERADOS

A) Los formatos de lectura específicos ingresan al programa principal por tarjeta leída.

CASO 1. El programa no tiene rutinas a las cuales se transmitan los formatos de lectura específicos.
CASO 2. Los formatos de lectura específicos se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. Los formatos de lectura específicos se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

B) Los formatos de lectura específicos están definidos por medio de una sentencia DATA o bien en el programa principal o en una subrutina BLOCK DATA. Las tiras de caracteres están precedidas por el código de conversión H o código HOLLERITH.

CASO 4. El programa no tiene rutinas a las cuales se transmitan los formatos de lectura específicos.
CASO 5. Los formatos de lectura específicos están incluídos en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. Los formatos de lectura específicos se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN LOS CASOS 1 Y 4.

Tres formatos de lectura están específicamente destinados a leer datos literales o números enteros o números flotantes a lo largo de todo el programa. Si estos formatos específicos ingresan al programa por tarjetas leídas con el formato convencional 103 (CASO 1), serán las tiras de caracteres

(4A4)
(2I3)
(3E14.6)

y ocuparán los vectores FLIT,FENT y FFLOT de dimensión 1, declarados REAL*8. Si estuvieran definidos en sentencias DATA (CASO 4), se los ha expresado en código

HOLLERITH (en vez de apóstrofos) y serán las tiras de caracteres

3H(4A,3H4)
3H(2I,3H3)
3H(3E,3H14.,3H6)

que ocuparán los vectores FLIT, FENT y FFLOT, todos REAL*4 y de dimensión 2 los dos primeros y de dimensión 3 el último.

En el programa se leerán los números enteros 2 y 5 con la sentencia READ(5,FENT) MIN,MAX. Con la sentencia READ(5,FLIT)PALABR se leerá la tira de caracteres CAL1CAL2CAL3CAL4, siendo PALABR un vector, REAL*4 y de dimensión 4. Por último, con la sentencia READ(5,FFLOT) (A(K),B(K),C(K),K=1,5) se leerán cinco grupos de números reales que llenarán los vectores A, B y C.

Después se efectuarán 4 cálculos distintos: según el valor de MAX se harán los cálculos del DO 1 y/o del DO 2 y según el valor del resultado de los mismos, se efectuarán los cálculos del DO 3 y/o del DO4. Para cada cálculo se escribirán los resultados con su nombre, con formatos convencionales de escritura, como prueba de que los formatos no convencionales de lectura funcionaron bien.

CAP.IV,PROGR.10,CASO 1. Los formatos de lectura específicos para literales, números enteros y números flotantes

(4A4)
(2I3)
(3E14.6)

ingresan al programa principal por tarjeta leída con formato convencional, a los vectores FLIT, FENT y FFLOT, REAL*8 y de dimensión 1. El programa no tiene rutinas a las cuales se transmitan los formatos de lectura específicos.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FLIT,FENT,FFLOT
      DIMENSION FLIT(1),FENT(1),FFLOT(1)
      DIMENSION A(5),B(5),C(5),PALABR(4)
      READ(5,103) FLIT,FENT,FFLOT
      READ(5,FENT) MIN,MAX
      READ(5,FLIT) PALABR
      READ(5,FFLOT) (A(K),B(K),C(K),K=1,5)
      SUM=0.
      IF(MAX.GE.(2*MIN+1)) GO TO 5
      DO 1 I=MIN,MAX
      SUM=SUM+(A(I)+B(I)/(A(I)-C(I)*B(I)))
    1 CONTINUE
      WRITE(6,100) PALABR(1),SUM
    5 DO 2 J=MIN,MAX
      SUM=SUM+SQRT(ABS(A(J)-B(J)))
    2 CONTINUE
      WRITE(6,100) PALABR(2),SUM
      IF(SUM) 6,6,7
```

```
    6 DO 3 K=MIN,MAX
      SUM=SUM-(C(K)-B(K)*A(K))
    3 CONTINUE
      WRITE(6,100) PALABR(3),SUM
    7 DO 4 L=MIN,MAX
      SUM=SUM+(A(L)*C(L))
    4 CONTINUE
      WRITE(6,100) PALABR(4),SUM
      SUMA=((SUM+A(1))*B(1))/C(5)
      WRITE(6,101) SUMA
      STOP
  100 FORMAT(1X,A4,3X,E14.6)
  101 FORMAT(1X,'SUMA=',E14.6)
  103 FORMAT(3A8)
      END
/*
//GO.SYSIN DD *
 (4A4)    (2I3)   (3E14.6)
  2  5
CAL1CAL2CAL3CAL4
  8.253410E 00 -3.141412E 00 -5.302145E 02
 -8.253410E 00  3.141412E 00  5.302145E 02
  8.874510E 00 -3.231052E 00 -5.231475E 02
  1.555222E-01 -3.000335E-02  0.896745E-05
  0.532710E 00 -2.854412E 00 -6.254145E-02
/*
//
```

CAP.IV,PROGR.10,CASO 2. Los formatos de lectura específicos para literales, números enteros y números flotantes

```
(4A4)
(2I3)
(3E14.6)
```

ingresan al programa principal por tarjeta leída con formato convencional, a los vectores FLIT, FENT y FFLOT, REAL*8, de dimensión 1 y se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FLIT,FENT,FFLOT
      COMMON/FORLEC/FLIT(1),FENT(1),FFLOT(1)
      READ(5,103) FLIT,FENT,FFLOT
      CALL CALCUL
      STOP
  103 FORMAT(3A8)
      END
C
      SUBROUTINE CALCUL
      REAL * 8 FLIT,FENT,FFLOT
      DIMENSION A(5),B(5),C(5),PALABR(4)
      COMMON/FORLEC/FLIT(1),FENT(1),FFLOT(1)
      READ(5,FENT) MIN,MAX
```

```
      READ(5,FLIT) PALABR
      READ(5,FFLOT) (A(K),B(K),C(K),K=1,5)
      SUM=0.
      IF(MAX.GE.(2*MIN+1)) GO TO 5
      DO 1 I=MIN,MAX
      SUM=SUM+(A(I)+B(I)/(A(I)-C(I)*B(I)))
    1 CONTINUE
      WRITE(6,100) PALABR(1),SUM
    5 DO 2 J=MIN,MAX
      SUM=SUM+SQRT(ABS(A(J)-B(J)))
    2 CONTINUE
      WRITE(6,100) PALABR(2),SUM
      IF(SUM) 6,6,7
    6 DO 3 K=MIN,MAX
      SUM=SUM-(C(K)-B(K)*A(K)
    3 CONTINUE
      WRITE(6,100) PALABR(3),SUM
    7 DO 4 L=MIN,MAX
      SUM=SUM+(A(L)*C(L))
    4 CONTINUE
      WRITE(6,100) PALABR(4),SUM
      SUMA=((SUM+A(1))*B(1))/C(5)
      WRITE(6,101) SUMA
      RETURN
  100 FORMAT(1X,A4,3X,E14.6)
  101 FORMAT(1X,'SUMA=',E14.6)
      END
/*
//GO.SYSIN DD *
 (4A4)   (2I3)  (3E14.6)
  2  5
CAL1CAL2CAL3CAL4
  8.253410E 00 -3.141412E 00 -5.302145E 02
 -8.253410E 00  3.141412E 00  5.302145E 02
  8.874510E 00 -3.231052E 00 -5.231475E 02
  1.555222E-01 -3.000335E-02  0.896745E-05
  0.532710E 00 -2.854412E 00 -6.254145E-02
/*
//
```

CAP.IV,PROGR.10,CASO 3. Los formatos de lectura específicos para literales, números enteros y números flotantes

```
(4A4)
(2I3)
(3E14.6)
```

ingresan al programa principal por tarjeta leída con formato convencional, a los vectores FLIT, FENT y FFLOT, REAL*8, de dimensión 1 y se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FLIT,FENT,FFLOT
      DIMENSION FLIT(1),FENT(1),FFLOT(1)
      READ(5,103) FLIT,FENT,FFLOT
```

```
      CALL CALCUL(FLIT,FENT,FFLOT)
      STOP
  103 FORMAT(3A8)
      END
C
      SUBROUTINE CALCUL(FLIT,FENT,FFLOT)
      REAL * 8 FLIT,FENT,FFLOT
      DIMENSION FLIT(1),FENT(1),FFLOT(1)
      DIMENSION A(5),B(5),C(5),PALABR(4)
      READ(5,FENT) MIN,MAX
      READ(5,FLIT) PALABR
      READ(5,FFLOT) (A(K),B(K),C(K),K=1,5)
      SUM=0.
      IF(MAX.GE.(2*MIN+1)) GO TO 5
      DO 1 I=MIN,MAX
    1 SUM=SUM+(A(I)+B(I)/(A(I)-C(I)*B(I)))
      WRITE(6,100) PALABR(1),SUM
    5 DO 2 J=MIN,MAX
    2 SUM=SUM+SQRT(ABS(A(J)-B(J)))
      WRITE(6,100) PALABR(2),SUM
      IF(SUM) 6,6,7
    6 DO 3 K=MIN,MAX
    3 SUM=SUM-(C(K)-B(K)*A(K))
      WRITE(6,100) PALABR(3),SUM
    7 DO 4 L=MIN,MAX
    4 SUM=SUM+(A(L)*C(L))
      WRITE(6,100) PALABR(4),SUM
      SUMA=((SUM+A(1))*B(1))/C(5)
      WRITE(6,101) SUMA
      RETURN
  100 FORMAT(1X,A4,3X,E14.6)
  101 FORMAT(1X,'SUMA=',E14.6)
      END
/*
//GO.SYSIN DD *
 (4A4)    (2I3)   (3E14.6)
  2  5
CAL1CAL2CAL3CAL4
  8.253410E 00 -3.141412E 00 -5.302145E 02
 -8.253410E 00  3.141412E 00  5.302145E 02
  8.874510E 00 -3.231052E 00 -5.231475E 02
  1.555222E-01 -3.000335E-02  0.896745E-05
  0.532710E 00 -2.854412E 00 -6.254145E-02
/*
//
```

CAP.IV,PROGR.10,CASO 4. Los formatos de lectura específicos para literales, números enteros y números flotantes

```
3H(4A,3H4 )
3H(2I,3H3 )
3H(3E,3H14.,3H6 )
```

están definidos en el programa principal por medio de sentencias DATA de nombres FLIT, FENT y FFLOT, utilizando el código HOLLERITH, todas REAL*4, de dimensión 2 las dos primeras y de dimensión 3 la última. El programa no tiene rutinas a las cuales se transmitan los formatos de lectura específicos.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION FLIT(2),FENT(2),FFLOT(3)
      DIMENSION A(5),B(5),C(5),PALABR(4)
      DATA FLIT/3H(4A,3H4 )/
      DATA FENT/3H(2I,3H3 )/
      DATA FFLOT/3H(3E,3H14.,3H6 )/
      READ(5,FENT) MIN,MAX
      READ(5,FLIT) PALABR
      READ(5,FFLOT) (A(K),B(K),C(K),K=1,5)
      SUM=0.
      IF(MAX.GE.(2*MIN+1)) GO TO 5
      DO 1 I=MIN,MAX
      SUM=SUM+(A(I)+B(I)/(A(I)-C(I)*B(I)))
    1 CONTINUE
      WRITE(6,100) PALABR(1),SUM
    5 DO 2 J=MIN,MAX
      SUM=SUM+SQRT(ABS(A(J)-B(J)))
    2 CONTINUE
      WRITE(6,100) PALABR(2),SUM
      IF(SUM) 6,6,7
    6 DO 3 K=MIN,MAX
      SUM=SUM-(C(K)-B(K)*A(K))
    3 CONTINUE
      WRITE(6,100) PALABR(3),SUM
    7 DO 4 L=MIN,MAX
      SUM=SUM+(A(L)*C(L))
    4 CONTINUE
      WRITE(6,100) PALABR(4),SUM
      SUMA=((SUM+A(1))*B(1))/C(5)
      WRITE(6,101) SUMA
      STOP
  100 FORMAT(1X,A4,3X,E14.6)
  101 FORMAT(1X,'SUMA=',E14.6)
      END
/*
//GO.SYSIN DD *
  2  5
CAL1CAL2CAL3CAL4
  8.253410E 00 -3.141412E 00 -5.302145E 02
 -8.253410E 00  3.141412E 00  5.302145E 02
  8.874510E 00 -3.231052E 00 -5.231475E 02
  1.555222E-01 -3.000335E-02  0.896745E-05
  0.532710E 00 -2.854412E 00 -6.254145E-02
/*
//
```

CAP.IV,PROGR.10,CASO 5. Los formatos de lectura específicos para literales, números enteros y números flotantes

```
3H(4A,3H4 )
3H(2I,3H3 )
3H(3E,3H14.,3H6 )
```

están definidos, utilizando el código HOLLERITH, por medio de sentencias DATA de nombres FLIT, FENT y FFLOT, incluídas en un BLOCK DATA; todas REAL*4, de dimen-

sión 2 las dos primeras y de dimensión 3 la última. Desde allí se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      CALL CALCUL
      STOP
      END
C
      BLOCK DATA
      COMMON/FORLEC/FLIT(2),FENT(2),FFLOT(3)
      DATA FLIT/3H(4A,3H4 )/
      DATA FENT/3H(2I,3H3 )/
      DATA FFLOT/3H(3E,3H14.,3H6 )/
      END
C
      SUBROUTINE CALCUL
      DIMENSION A(5),B(5),C(5),PALABR(4)
      COMMON/FORLEC/FLIT(2),FENT(2),FFLOT(3)
      READ(5,FENT) MIN,MAX
      READ(5,FLIT) PALABR
      READ(5,FFLOT) (A(K),B(K),C(K),K=1,5)
      SUM=0.
      IF(MAX.GE.(2*MIN+1)) GO TO 5
      DO 1 I=MIN,MAX
      SUM=SUM+(A(I)+B(I)/(A(I)-C(I)*B(I)))
    1 CONTINUE
      WRITE(6,100) PALABR(1),SUM
    5 DO 2 J=MIN,MAX
      SUM=SUM+SQRT(ABS(A(J)-B(J)))
    2 CONTINUE
      WRITE(6,100) PALABR(2),SUM
      IF(SUM) 6,6,7
    6 DO 3 K=MIN,MAX
      SUM=SUM-(C(K)-B(K)*A(K))
    3 CONTINUE
      WRITE(6,100) PALABR(3),SUM
    7 DO 4 L=MIN,MAX
      SUM=SUM+(A(L)*C(L))
    4 CONTINUE
      WRITE(6,100) PALABR(4),SUM
      SUMA=((SUM+A(1))*B(1))/C(5)
      WRITE(6,101) SUMA
      RETURN
  100 FORMAT(1X,A4,3X,E14.6)
  101 FORMAT(1X,'SUMA='E14.6)
      END
/*
//GO.SYSIN DD *
  2  5
CAL1CAL2CAL3CAL4
  8.253410E 00 -3.141412E 00 -5.302145E 02
 -8.253410E 00  3.141412E 00  5.302145E 02
  8.874510E 00 -3.231052E 00 -5.231475E 02
  1.555222E-01 -3.000335E-02  0.896745E-05
```

```
  0.532710E 00 -2.854412E 00 -6.254145E-02
/*
//
```

CAP.IV,PROGR.10,CASO 6. Los formatos de lectura específicos para literales, números enteros y números flotantes

```
3H(4A,3H4 )
3H(2I,3H3 )
3H(3E,3H14.,3H6 )
```

están definidos en el programa principal por medio de sentencias DATA de nombre FLIT, FENT y FFLOT, utilizando el código HOLLERITH; todas REAL*4, de dimensión 2 las dos primeras y de dimensión 3 la última. Desde allí se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION FLIT(2),FENT(2),FFLOT(3)
      DATA FLIT/3H(4A,3H4 )/
      DATA FENT/3H(2I,3H3 )/
      DATA FFLOT/3H(3E,3H14.,3H6 )/
      CALL CALCUL(FLIT,FENT,FFLOT)
      STOP
      END
C
      SUBROUTINE CALCUL(FLIT,FENT,FFLOT)
      DIMENSION FLIT(2),FENT(2),FFLOT(3)
      DIMENSION A(5),B(5),C(5),PALABR(4)
      READ(5,FENT) MIN,MAX
      READ(5,FLIT) PALABR
      READ(5,FFLOT) (A(K),B(K),C(K),K=1,5)
      SUM=0.
      IF(MAX.GE.(2*MIN+1)) GO TO 5
      DO 1 I=MIN,MAX
      SUM=SUM+(A(I)+B(I)/(A(I)-C(I)*B(I)))
    1 CONTINUE
      WRITE(6,100) PALABR(1),SUM
    5 DO 2 J=MIN,MAX
      SUM=SUM+SQRT(ABS(A(J)-B(J)))
    2 CONTINUE
      WRITE(6,100) PALABR(2),SUM
      IF(SUM) 6,6,7
    6 DO 3 K=MIN,MAX
      SUM=SUM-(C(K)-B(K)*A(K))
    3 CONTINUE
      WRITE(6,100) PALABR(3),SUM
    7 DO 4 L=MIN,MAX
      SUM=SUM+(A(L)*C(L))
    4 CONTINUE
      WRITE(6,100) PALABR(4),SUM
      SUMA=((SUM+A(1))*B(1))/C(5)
      WRITE(6,101) SUMA
      RETURN
```

```
  100 FORMAT(1X,A4,3X,E14.6)
  101 FORMAT(1X,'SUMA=',E14.6)
      END
/*
//GO.SYSIN DD *
  2  5
CAL1CAL2CAL3CAL4
  8.253410E 00 -3.141412E 00 -5.302145E 02
 -8.253410E 00  3.141412E 00  5.302145E 02
  8.874510E 00 -3.231052E 00 -5.231475E 02
  1.555222E-01 -3.000335E-02  0.896745E-05
  0.532710E 00 -2.854412E 00 -6.254145E-02
/*
//
```

CAPITULO V

FORMATOS DE ESCRITURA QUE SE ELIGEN DURANTE LA EJECUCION DEL PROGRAMA EXPRESADOS UTILIZANDO APOSTROFOS O CODIGO HOLLERITH O AMBOS SISTEMAS COMBINADOS

PROGRAMA 11.

OBJETO: El programa tiene preparados 3 formatos de escritura de inscripciones. Según el resultado numérico y la consiguiente decisión de sentencias IF, elegirá cuál es la inscripción que será escrita. Ya sea que los formatos ingresen al programa por tarjeta leída o estén definidos en sentencias DATA, estarán expresados utilizando exclusivamente el código de apóstrofos.

CASOS CONSIDERADOS

A) Los formatos de escritura preparados ingresan al programa principal por tarjeta leída, con las tiras de caracteres encerradas entre apóstrofos.

CASO 1. El programa no tiene rutinas a las cuales se transmitan los formatos de escritura preparados.
CASO 2. Los formatos de escritura preparados se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. Los formatos de escritura preparados se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

B) Los formatos de escritura preparados están definidos por medio de una sentencia DATA o bien en el programa principal o en una subrutina BLOCK DATA. Las tiras de caracteres se encierran entre apóstrofos.

CASO 4. El programa no tiene rutinas a las cuales se transmitan los formatos de escritura preparados.
CASO 5. Los formatos de escritura preparados están incluídos en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. Los formatos de escritura preparados se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

Los formatos de escritura ya preparados que posee el programa son los 3 siguientes:

```
(1X,' FIELD1=',I2,' PARTE2=',F5.2,' FILA1=',I2)
(1X,' FIELD2=',I2,' PARTE4=',F5.2,' FILA3=',I2)
(1X,' FIELD3=',I2,' PARTE6=',F5.2,' FILA5=',I2)
```

Ingresarán por tarjeta leída con el formato convencional 100, a los vectores FORM1, FORM2 y FORM3 de dimensión 6 y declarados REAL*8. Después de realizar varios cálculos el programa llegará a un grupo de 3 sentencias IF en donde, según cual de ellas se cumpla, escribirá con alguno de los formatos que tiene preparados.

b) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 4.

Los formatos de escritura ya preparados que posee el programa están definidos en las siguientes sentencias DATA:

```
      DATA FORM1/'  (1X,  ','''FI1=''',',I2,3X, ','''PR2=''',' ,F5.2, ',
     1  '  3X,   ','''FL1=''','  ,I2)  '/
      DATA FORM2/'  (1X,  ','''FI2=''',',I2,3X, ','''PR4=''',' ,F5.2, ',
     1  '  3X,   ','''FL3=''','  ,I2)  '/
      DATA FORM3/'  (1X,  ','''FI3=''',',I2,3X, ','''PR6=''',' ,F5.2, ',
     1  '  3X,   ','''FL5=''','  ,I2)  '/
```

Debe distinguirse entre los apóstrofos propios de la sentencia DATA y los dobles apóstrofos que especifican la inscripción que se espera imprimir. En el primer DATA se pueden localizar los apóstrofos que encierran grupos de 8 caracteres, separados por una coma y que aparecen 8 veces lo cual se corresponde con el vector FORM1 declarado REAL*8 y de dimensión 8. Los apóstrofos que definen las leyendas a escribir son ''FI1='',''PR2='' y ''FL1=''. Lo mismo se advierte en los DATA siguientes que son totalmente análogos.

Después de realizar varios cálculos el programa llegará a un grupo de 3 sentencias IF en donde, según cual de ellas se cumpla, escribirá con alguno de los formatos que tiene preparados.

CAP.V,PROGR.11,CASO 1. Los formatos de escritura preparados ingresan al programa principal por tarjeta leída con el formato convencional 100, a los vectores FORM1, FORM2 y FORM3, REAL*8 y de dimensión 6. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. El programa no tiene rutinas a las cuales se transmitan los formatos preparados.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3
      DIMENSION FORM1(6),FORM2(6),FORM3(6)
      READ(5,100) FORM1
      READ(5,100) FORM2
      READ(5,100) FORM3
      READ(5,101) J,K,L,B,C,D
      I=K-L+J
      A=B+C-D
      M=(K+3)*L
      IF(I.GT.5.AND.B.LT.0.009) WRITE(6,FORM1) I,A,M
      IF(I.EQ.1.AND.B.GT.0.001) WRITE(6,FORM2) I,A,M
      IF(I.LT.1.AND.B.LT.0.0001) WRITE(6,FORM3) I,A,M
      STOP
  100 FORMAT(6A8)
  101 FORMAT(3I3,3F7.2)
      END
/*
//GO.SYSIN DD *
```

```
(1X,' FIELD1=',I2,' PARTE2=',F5.2,' FILA1=',I2)
(1X,' FIELD2=',I2,' PARTE4=',F5.2,' FILA3=',I2)
(1X,' FIELD3=',I2,' PARTE6=',F5.2,' FILA5=',I2)
  3 -1  1  25.32 -14.23  10.05
/*
//
```

CAP.V,PROGR.11,CASO 2. Los formatos de escritura preparados ingresan al programa principal por tarjeta leída con el formato convencional 100, a los vectores FORM1, FORM2 y FORM3, REAL*8 y de dimensión 6. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos preparados se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3
      COMMON/FORESC/FORM1(6),FORM2(6),FORM3(6)
      READ(5,100) FORM1
      READ(5,100) FORM2
      READ(5,100) FORM3
      READ(5,101) J,K,L,B,C,D
      CALL CALCUL(J,K,L,B,C,D)
      STOP
  100 FORMAT(6A8)
  101 FORMAT(3I3,3F7.2)
      END
C
      SUBROUTINE CALCUL(J,K,L,B,C,D)
      REAL * 8 FORM1,FORM2,FORM3
      COMMON/FORESC/FORM1(6),FORM2(6),FORM3(6)
      I=K-L+J
      A=B+C-D
      M=(K+3)*L
      IF(I.GT.5.AND.B.LT.0.009) WRITE(6,FORM1) I,A,M
      IF(I.EQ.1.AND.B.GT.0.001) WRITE(6,FORM2) I,A,M
      IF(I.LT.1.AND.B.LT.0.0001) WRITE(6,FORM3) I,A,M
      RETURN
      END
/*
//GO.SYSIN DD *
(1X,' FIELD1=',I2,' PARTE2=',F5.2,' FILA1=',I2)
(1X,' FIELD2=',I2,' PARTE4=',F5.2,' FILA3=',I2)
(1X,' FIELD3=',I2,' PARTE6=',F5.2,' FILA5=' I2)
  3 -1  1  25.32 -14.23  10.05
/*
//
```

CAP.V,PROGR.11,CASO 3. Los formatos de escritura preparados ingresan al programa principal por tarjeta leída con el formato convencional 100, a los vectores FORM1, FORM2 y FORM3, REAL*8 y de dimensión 6. En algún momento de la ejecución

se elegirá el formato con el cual se va a escribir. Los formatos preparados se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3
      DIMENSION FORM1(6),FORM2(6),FORM3(6)
      READ(5,100) FORM1
      READ(5,100) FORM2
      READ(5,100) FORM3
      READ(5,101) J,K,L,B,C,D
      CALL CALCUL(FORM1,FORM2,FORM3,J,K,L,B,C,D)
      STOP
  100 FORMAT(6A8)
  101 FORMAT(3I3,3F7.2)
      END
C
      SUBROUTINE CALCUL(FORM1,FORM2,FORM3,J,K,L,B,C,D)
      DIMENSION FORM1(6),FORM2(6),FORM3(6)
      I=K-L+J
      A=B+C-D
      M=(K+3)*L
      IF(I.GT.5.AND.B.LT.0.009) WRITE(6,FORM1) I,A,M
      IF(I.EQ.1.AND.B.GT.0.001) WRITE(6,FORM2) I,A,M
      IF(I.LT.1.AND.B.LT.0.0001) WRITE(6,FORM3) I,A,M
      RETURN
      END
/*
//GO.SYSIN DD *
(1X,' FIELD1=',I2,' PARTE2=',F5.2,' FILA1=',I2)
(1X,' FIELD2=',I2,' PARTE4=',F5.2,' FILA3=',I2)
(1X,' FIELD3=',I2,' PARTE6=',F5.2,' FILA5=',I2)
  3 -1  1  25.32 -14.23  10.05
/*
//
```

CAP.V,PROGR.11,CASO 4. Los formatos de escritura preparados están definidos en el programa principal por medio de sentencias DATA de nombres FORM1, FORM2 y FORM3, REAL*8 y de dimensión 8. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. El programa no tiene rutinas a las cuales se transmitan los formatos preparados.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3
      DIMENSION FORM1(8),FORM2(8),FORM3(8)
      DATA FORM1/'  (1X,  ','''FI1=''',',I2,3X, ','''PR2=''',' ,F5.2, ',
     1  '  3X,   ','''FL1=''','  ,I2)  '/
      DATA FORM2/'  (1X,  ','''FI2=''',',I2,3X, ','''PR4=''',' ,F5.2, ',
     1  '  3X,   ','''FL3=''','  ,I2)  '/
```

```
      DATA FORM3/'  (1X,  ',''''FI3=''',',I2,3X, ',''''PR6=''',' ,F5.2, ',
     1  '  3X,   ',''''FL5=''','  ,I2)  '/
      READ(5,100) J,K,L,B,C,D
      I=K-L+J
      A=B+C-D
      M=(K+3)*L
      IF(I.GT.5.AND.B.LT.0.009) WRITE(6,FORM1) I,A,M
      IF(I.EQ.1.AND.B.GT.0.001) WRITE(6,FORM2) I,A,M
      IF(I.LT.1.AND.B.LT.0.0001)  WRITE(6,FORM3) I,A,M
      STOP
  100 FORMAT(3I3,3F7.2)
      END
/*
//GO.SYSIN DD *
  3 -1  1  25.32 -14.23  10.05
/*
//
```

CAP.V,PROGR.11,CASO 5. Los formatos de escritura preparados están definidos por medio de sentencias DATA de nombres FORM1, FORM2 y FORM3, REAL*8 y de dimensión 8, incluídas en un BLOCK DATA. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos de escritura preparados se transmiten a la SUBROUTINE CALCUL por medio de un COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) J,K,L,B,C,D
      CALL CALCUL(J,K,L,B,C,D)
      STOP
  100 FORMAT(3I3,3F7.2)
      END
C
      BLOCK DATA
      REAL * 8 FORM1,FORM2,FORM3
      COMMON/FORESC/FORM1(8),FORM2(8),FORM3(8)
      DATA FORM1/'  (1X,  ',''''FI1=''',',I2,3X, ',''''PR2=''',' ,F5.2, ',
     1  '  3X,   ',''''FL1=''',' ,I2)  '/
      DATA FORM2/'  (1X,  ',''''FI2=''',',I2,3X, ',''''PR4=''',' ,F5.2, ',
     1  '  3X,   ',''''FL3=''',' ,I2)  '/
      DATA FORM3/'  (1X,  ',''''FI3=''',',I2,3X, ',''''PR6=''',' ,F5.2, ',
     1  '  3X,   ',''''FL5=''',' ,I2)  '/
      END
C
      SUBROUTINE CALCUL(J,K,L,B,C,D)
      REAL * 8 FORM1,FORM2,FORM3
      COMMON/FORESC/FORM1(8),FORM2(8),FORM3(8)
      I=K-L+J
      A=B+C-D
      M=(K+3)*L
      IF(I.GT.5.AND.B.LT.0.009) WRITE(6,FORM1) I,A,M
      IF(I.EQ.1.AND.B.GT.0.001) WRITE(6,FORM2) I,A,M
      IF(I.LT.1.AND.B.LT.0.0001) WRITE(6,FORM3) I,A,M
      RETURN
```

```
      END
/*
//GO.SYSIN DD *
  3 -1  1  25.32 -14.23  10.05
/*
//
```

CAP.V,PROGR.11,CASO 6. Los formatos de escritura preparados están definidos en el programa principal por medio de sentencias DATA de nombres FORM1, FORM2 y FORM3, REAL*8 y de dimensión 8. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos de escritura preparados se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3
      DIMENSION FORM1(8),FORM2(8),FORM3(8)
      DATA FORM1/'  (1X,  ',''''FI1='''',',I2,3X, ',''''PR2='''',' ,F5.2, ',
     1  '  3X,  ',''''FL1='''','  ,I2)  '/
      DATA FORM2/'  (1X,  ',''''FI2='''',',I2,3X, ',''''PR4='''',' ,F5.2, ',
     1  '  3X,  ',''''FL3='''','  ,I2)  '/
      DATA FORM3/'  (1X,  ',''''FI3='''',',I2,3X, ',''''PR6='''',' ,F5.2, ',
     1  '  3X,  ',''''FL5='''','  ,I2)  '/
      READ(5,100) J,K,L,B,C,D
      CALL CALCUL(FORM1,FORM2,FORM3,J,K,L,B,C,D)
      STOP
  100 FORMAT(3I3,3F7.2)
      END
C
      SUBROUTINE CALCUL(FORM1,FORM2,FORM3,J,K,L,B,C,D)
      DIMENSION FORM1(8),FORM2(8),FORM3(8)
      I=K-L+J
      A=B+C-D
      M=(K+3)*L
      IF(I.GT.5.AND.B.LT.0.009) WRITE(6,FORM1) I,A,M
      IF(I.EQ.1.AND.B.GT.0.001) WRITE(6,FORM2) I,A,M
      IF(I.LT.1.AND.B.LT.0.0001) WRITE(6,FORM3) I,A,M
      RETURN
      END
/*
//GO.SYSIN DD *
  3 -1  1  25.32 -14.23  10.05
/*
//
```

PROGRAMA 12.

OBJETO: El programa tiene preparados 2 formatos de escritura de inscripciones. Según el resultado numérico y la consiguiente decisión de sentencias IF, elegirá cual es la inscripción que será escrita. Ya sea que los formatos ingresen al programa por tarjeta leída o estén definidos en sentencias DATA, están expresados utilizando exclusivamente el código HOLLERITH.

CASOS CONSIDERADOS

A) Los formatos de escritura preparados ingresan al programa principal por tarjeta leída, con las tiras de caracteres precedidas por el código HOLLERITH.

CASO 1. El programa no tiene rutinas a las cuales se transmitan los formatos de escritura preparados.
CASO 2. Los formatos de escritura preparados se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. Los formatos de escritura preparados se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

B) Los formatos de escritura preparados están definidos por medio de una sentencia DATA o bien en el programa principal o en una subrutina BLOCK DATA. Las tiras de caracteres están precedidas por el código HOLLERITH.

CASO 4. El programa no tiene rutinas a las cuales se transmitan los formatos de escritura preparados.
CASO 5. Los formatos de escritura preparados están incluídos en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. Los formatos de escritura preparados se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

Los formatos de escritura ya preparados que posee el programa son los 2 siguientes:

```
(1X,29H ANTAR MENOR O IGUAL QUE 0.3.,2X,3H A=,F5.2)
(1X,21H ANTAR MAYOR QUE 0.3.,2X,3H A=,F5.2)
```

Ingresarán por tarjeta leída con los formatos convencionales 101 y 102 respectivamente, a los vectores FORM1 y FORM2 de dimensiones 7 y 6 y declarados REAL*8.

Después de calcular la variable ANTAR una sentencia IF decidirá cuál de los formatos preparados es el que se elegirá para escribir.

b) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 4.

Los formatos de escritura ya preparados que posee el programa están definidos en las siguientes sentencias DATA:

```
      DATA FORM1/8H(1X,29H ,8HANTAR ME,8HNOR O IG,8HUAL QUE ,8HO.3.,2X,,
     1  8H3H A=,F5,8H.2)     /
      DATA FORM2/8H(1X,21H ,8HANTAR MA,8HYOR QUE ,8HO.3.,2X,,8H3H A=,F5,
     1  8H.2)     /
```

Debe distinguirse entre las H propias de la sentencia DATA y las H que especifican la inscripción que se espera imprimir. En el primer DATA el grupo 8H aparece 7 veces, seguido siempre por 8 caracteres y separado del siguiente por una coma, lo cual se corresponde con el vector FORM1 de dimensión 7 y declarado REAL*8. Los grupos 29H y 3H son los que definen las leyendas a escribir.

De la misma manera en el segundo DATA aparecen las H propias de la sentencia DATA representadas por el grupo 8H repetido 6 veces y seguido de 8 caracteres, en consonancia con el vector FORM2(6) y REAL*8. Los grupos 21H y 3H son los que definen las leyendas a escribir. Después de calcular la variable ANTAR, una sentencia IF decidirá cuál de los dos formatos preparados es el que se elegirá para escribir.

CAP.V,PROGR.12,CASO 1. Los formatos de escritura preparados ingresan al programa principal por tarjeta leída con los formatos convencionales 101 y 102, a los vectores FORM1 y FORM2, REAL*8 y de dimensión 7 y 6 respectivamente. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. El programa no tiene rutinas a las cuales se transmitan los formatos preparados.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2
      DIMENSION FORM1(7),FORM2(6)
      READ(5,101) FORM1
      READ(5,102) FORM2
      DO 3 I=1,2
      READ(5,100) B,C,D
      ANTAR=(B+C)*SQRT(ABS(B-C+D))
      IF(ANTAR.LE.0.3) GO TO 1
      A=ANTAR-(0.01+D)
      WRITE(6,FORM2) A
      GO TO 2
    1 A=ANTAR+(0.01-D)
      WRITE(6,FORM1) A
    3 CONTINUE
    2 STOP
  100 FORMAT(3F7.2)
  101 FORMAT(7A8)
  102 FORMAT(6A8)
```

```
      END
/*
//GO.SYSIN DD *
(1X,29H ANTAR MENOR O IGUAL QUE 0.3.,2X,3H A=,F5.2)
(1X,21H ANTAR MAYOR QUE 0.3.,2X,3H A=,F5.2)
   0.34  -0.78   0.17
  -0.34   0.78   0.17
/*
//
```

CAP.V,PROGR.12,CASO 2. Los formatos de escritura preparados ingresan al programa principal por tarjeta leída con los formatos convencionales 101 y 102, a los vectores FORM1 y FORM2, REAL*8 y de dimensión 7 y 6 respectivamente. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos preparados se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2
      COMMON/FORESC/FORM1(7),FORM2(6)
      READ(5,101) FORM1
      READ(5,102) FORM2
      CALL CALCUL
      STOP
  101 FORMAT(7A8)
  102 FORMAT(6A8)
      END
C
      SUBROUTINE CALCUL
      REAL * 8 FORM1,FORM2
      COMMON/FORESC/FORM1(7),FORM2(6)
      DO 3 I=1,2
      READ(5,100) B,C,D
      ANTAR=(B+C)*SQRT(ABS(B-C+D))
      IF(ANTAR.LE.0.3) GO TO 1
      A=ANTAR-(0.01+D)
      WRITE(6,FORM2) A
      GO TO 2
    1 A=ANTAR+(0.01-D)
      WRITE(6,FORM1) A
    3 CONTINUE
    2 RETURN
  100 FORMAT(3F7.2)
      END
/*
//GO.SYSIN DD *
(1X,29H ANTAR MENOR O IGUAL QUE 0.3.,2X,3H A=,F5.2)
(1X,21H ANTAR MAYOR QUE 0.3.,2X,3H A=,F5.2)
   0.34  -0.78   0.17
  -0.34   0.78   0.17
/*
//
```

CAP.V,PROGR.12,CASO 3. Los formatos de escritura preparados ingresan al programa principal por tarjeta leída con los formatos convencionales 101 y 102, a los vectores FORM1 y FORM2, REAL*8 y de dimensión 7 y 6 respectivamente. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos preparados se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2
      DIMENSION FORM1(7),FORM2(6)
      READ(5,101) FORM1
      READ(5,102) FORM2
      CALL CALCUL(FORM1,FORM2)
      STOP
  101 FORMAT(7A8)
  102 FORMAT(6A8)
      END
C
      SUBROUTINE CALCUL(FORM1,FORM2)
      DIMENSION FORM1(7),FORM2(6)
      DO 3 I=1,2
      READ(5,100) B,C,D
      ANTAR=(B+C)*SQRT(ABS(B-C+D))
      IF(ANTAR.LE.0.3) GO TO 1
      A =ANTAR-(0.01+D)
      WRITE(6,FORM2) A
      GO TO 2
    1 A=ANTAR+(0.01-D)
      WRITE(6,FORM1) A
    3 CONTINUE
    2 RETURN
  100 FORMAT(3F7.2)
      END
/*
//GO.SYSIN DD *
(1X,29H ANTAR MENOR O IGUAL QUE 0.3.,2X,3H A=,F5.2)
(1X,21H ANTAR MAYOR QUE 0.3.,2X,3H A=,F5.2)
   0.34  -0.78   0.17
  -0.34   0.78   0.17
/*
//
```

CAP.V,PROGR.12,CASO 4. Los formatos de escritura preparados están definidos en el programa principal por medio de sentencias DATA de nombres FORM1 y FORM2, REAL*8 y de dimensión 7 y 6 respectivamente. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. El programa no tiene rutinas a las cuales se transmitan los formatos preparados.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
```

```
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2
      DIMENSION FORM1(7),FORM2(6)
      DATA FORM1/8H(1X,29H ,8HANTAR ME,8HNOR O IG,8HUAL QUE ,8H0.3.,2X,,
     1  8H3H A=,F5,8H.2)     /
      DATA FORM2/8H(1X,21H ,8HANTAR MA,8HYOR QUE ,8H0.3.,2X,,8H3H A=,F5,
     1  8H.2)     /
      DO 3 I=1,2
      READ(5,100) B,C,D
      ANTAR=(B+C)*SQRT(ABS(B-C+D))
      IF(ANTAR.LE.0.3) GO TO 1
      A=ANTAR-(0.01+D)
      WRITE(6,FORM2) A
      GO TO 2
    1 A=ANTAR+(0.01-D)
      WRITE(6,FORM1) A
    3 CONTINUE
    2 STOP
  100 FORMAT(3F7.2)
      END
/*
//GO.SYSIN DD *
   0.34  -0.78   0.17
  -0.34   0.78   0.17
/*
//
```

CAP.V,PROGR.12,CASO 5. Los formatos de escritura preparados están definidos por medio de sentencias DATA de nombres FORM1 y FORM2, REAL*8, de dimensión 7 y 6 respectivamente, incluídas en un BLOCK DATA. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos de escritura preparados se transmiten a la SUBROUTINE CALCUL por medio de un COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      CALL CALCUL
      STOP
      END
C
      BLOCK DATA
      REAL * 8 FORM1,FORM2
      COMMON/FORESC/FORM1(7),FORM2(6)
      DATA FORM1/8H(1X,29H ,8HANTAR ME,8HNOR O IG,8HUAL QUE ,8H0.3.,2X,,
     1  8H3H A=,F5,8H.2)     /
      DATA FORM2/8H(1X,21H ,8HANTAR MA,8HYOR QUE ,8H0.3.,2X,,8H3H A=,F5,
     1  8H.2)     /
      END
C
      SUBROUTINE CALCUL
      REAL * 8 FORM1,FORM2
      COMMON/FORESC/FORM1(7),FORM2(6)
      DO 3 I=1,2
      READ(5,100) B,C,D
```

```
      ANTAR=(B+C)*SQRT(ABS(B-C+D))
      IF(ANTAR.LE.0.3) GO TO 1
      A=ANTAR-(0.01+D)
      WRITE(6,FORM2) A
      GO TO 2
    1 A=ANTAR+(0.01-D)
      WRITE(6,FORM1) A
    3 CONTINUE
    2 RETURN
  100 FORMAT(3F7.2)
      END
/*
//GO.SYSIN DD *
   0.34   -0.78    0.17
  -0.34    0.78    0.17
/*
//
```

CAP.V,PROGR.12,CASO 6. Los formatos de escritura preparados están definidos en el programa principal por medio de sentencias DATA de nombres FORM1 y FORM2, REAL*8 y de dimensión 7 y 6 respectivamente. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos de escritura preparados se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB .........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2
      DIMENSION FORM1(7),FORM2(6)
      DATA FORM1/8H(1X,29H ,8HANTAR ME,8HNOR O IG,8HUAL QUE ,8H0.3.,2X,,
     1  8H3H A=,F5,8H.2)     /
      DATA FORM2/8H(1X,21H ,8HANTAR MA,8HYOR QUE ,8H0.3.,2X,,8H3H A=,F5,
     1  8H.2)     /
      CALL CALCUL(FORM1,FORM2)
      STOP
      END
C
      SUBROUTINE CALCUL(FORM1,FORM2)
      DIMENSION FORM1(7),FORM2(6)
      DO 3 I=1,2
      READ(5,100) B,C,D
      ANTAR=(B+C)*SQRT(ABS(B-C+D))
      IF(ANTAR.LE.0.3) GO TO 1
      A=ANTAR-(0.01+D)
      WRITE(6,FORM2) A
      GO TO 2
    1 A=ANTAR+(0.01-D)
      WRITE(6,FORM1) A
    3 CONTINUE
    2 RETURN
  100 FORMAT(3F7.2)
      END
```

```
/*
//GO.SYSIN DD *
  0.34  -0.78   0.17
 -0.34   0.78   0.17
/*
//
```

PROGRAMA 13.

OBJETO: El programa tiene preparados 4 formatos de escritura de inscripciones. Según el resultado numérico y la consiguiente decisión de sentencias IF, elegirá cuál es la inscripción que será escrita. Si los formatos ingresan al programa por tarjeta leída, están expresados utilizando exclusivamente el código de apóstrofos, pero si los formatos están definidos en sentencias DATA, están expresados utilizando los códigos de apóstrofos y HOLLERITH combinados.

CASOS CONSIDERADOS

A) Los formatos de escritura preparados ingresan al programa principal por tarjeta leída, con las tiras de caracteres encerradas entre apóstrofos.

CASO 1. El programa no tiene rutinas a las cuales se transmitan los formatos de escritura preparados.
CASO 2. Los formatos de escritura preparados se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. Los formatos de escritura preparados se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

B) Los formatos de escritura preparados están definidos por medio de una sentencia DATA o bien en el programa principal o en una subrutina BLOCK DATA. Los formatos están expresados utilizando los códigos de apóstrofos y HOLLERITH combinados.

CASO 4. El programa no tiene rutinas a las cuales se transmitan los formatos de escritura preparados.
CASO 5. Los formatos de escritura preparados están incluídos en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. Los formatos de escritura preparados se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

Los formatos de escritura ya preparados que posee el programa son los 4 siguientes:

```
(1X,' VELOCIDAD NO CORREGIDA',2X,E14.6)
(1X,' VELOCIDAD CORREGIDA POR EXCESO',2X,E14.6)
(1X,' VELOCIDAD CORREGIDA POR DEFECTO',2X,E14.6)
(1X,' CALCULO 2 VELOCIDADES. NO TENERLAS EN CUENTA')
```

e ingresan por tarjeta leída con los formatos convencionales 100, 101, 101 y 102

a los vectores FORM1, FORM2, FORM3 y FORM4 de dimensión 5, 6, 6 y 7 respectivamente, declarados todos REAL*8. Después de realizar varios cálculos el programa llegará a un grupo de 4 sentencias IF en donde, según cual de ellas se cumpla, escribirá con alguno de los formatos que tiene preparados.

b) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 4.

Los formatos de escritura ya preparados que posee el programa están definidos en las siguientes sentencias DATA, declaradas REAL*8 y de dimensión 5, 6, 7 y 7 respectivamente:

```
      DATA FORM1/'(1X,23H ','VELOCIDA','D NO COR','REGIDA,2','X,E14.6)'/
      DATA FORM2/'(1X,31H ','VELOCIDA','D CORREG','IDA POR ','EXCESO,2',
     1  'X,E14.6)'/
      DATA FORM3/'(1X,32H ','VELOCIDA','D CORREG','IDA POR ','DEFECTO,',
     1  '2X,E14.6',')        '/
      DATA FORM4/'(1X,43HC','ALCULO 2',' VELOCID','ADES.NO ','TENERLAS',
     1  ' EN CUEN','TA)      '/
```

Debe distinguirse entre los apóstrofos propios de la sentencia DATA y las H que especifican la inscripción que se espera imprimir. En el primer DATA se advierten los apóstrofos que encierran grupos de 8 caracteres separados por una coma y que aparecen 5 veces lo cual se corresponde con el vector FORM1 declarado REAL*8 y de dimensión 5; el grupo 23H es el que define la leyenda a escribir.

En el DATA siguiente los apóstrofos encerrando 8 caracteres aparecen 6 veces lo cual se corresponde con el vector FORM2(6) y REAL*8; el grupo 31H es el que define la leyenda a escribir. En el tercer DATA los apóstrofos encerrando 8 caracteres aparecen 7 veces lo cual se corresponde con el vector FORM3(7) y REAL*8; el grupo 32H es el que define la leyenda a escribir. En el último DATA los apóstrofos encerrando 8 caracteres aparecen 7 veces en consonancia con el vector FORM4(7) y REAL*8; el grupo 43H es el que define la leyenda a escribir.

Después de realizar varios cálculos el programa llegará a un grupo de 4 sentencias IF en donde, según cual de ellas se cumpla, escribirá con alguno de los formatos que tiene preparados.

CAP.V,PROGR.13,CASO 1. Los formatos de escritura preparados ingresan al programa principal por tarjetas leídas con los formatos convencionales 100,101,101 y 102, a los vectores FORM1, FORM2, FORM3 y FORM4, declarados REAL*8. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. El programa no tiene rutinas a las cuales se transmitan los formatos preparados.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3,FORM4
      DIMENSION FORM1(5),FORM2(6),FORM3(6),FORM4(7)
      READ(5,100) FORM1
```

```
      READ(5,101) FORM2
      READ(5,101) FORM3
      READ(5,102) FORM4
      READ(5,103) INDICE,MODULO,X,Y,Z
      K=INDICE*MODULO
      V=((X+Y)/Z)+FLOAT(K)
      IF(MODULO.LT.0) V=V+0.173
      IF(INDICE.LT.0) V=V-0.178
      IF(MODULO.GE.0.AND.INDICE.GE.0) WRITE(6,FORM1) V
      IF(MODULO.GE.0.AND.INDICE.LT.0) WRITE(6,FORM2) V
      IF(MODULO.LT.0 AND.INDICE.GE.0) WRITE(6,FORM3) V
      IF(MODULO.LT.0.AND.INDICE.LT.0) WRITE(6,FORM4) V
      STOP
  100 FORMAT(5A8)
  101 FORMAT(6A8)
  102 FORMAT(7A8)
  103 FORMAT(2I3,3F5.3)
      END
/*
//GO.SYSIN DD *
(1X,' VELOCIDAD NO CORREGIDA',2X,E14.6)
(1X,' VELOCIDAD CORREGIDA POR EXCESO',2X,E14.6)
(1X,' VELOCIDAD CORREGIDA POR DEFECTO',2X,E14.6)
(1X,' CALCULO 2 VELOCIDADES.NO TENERLAS EN CUENTA')
  2  3 .321 .745-.012
/*
//
```

CAP.V,PROGR.13,CASO 2. Los formatos de escritura preparados ingresan al programa principal por tarjetas leídas con los formatos convencionales 100, 101, 101 y 102, a los vectores FORM1, FORM2,FORM3 y FORM4, declarados REAL*8 y de dimensión 5, 6, 6 y 7 respectivamente. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos preparados se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3,FORM4
      COMMON/FORESC/FORM1(5),FORM2(6),FORM3(6),FORM4(7)
      READ(5,100) FORM1
      READ(5,101) FORM2
      READ(5,101) FORM3
      READ(5,102) FORM4
      READ(5,103) INDICE,MODULO,X,Y,Z
      CALL CALCUL(INDICE,MODULO,X,Y,Z)
      STOP
  100 FORMAT(5A8)
  101 FORMAT(6A8)
  102 FORMAT(7A8)
  103 FORMAT(2I3,3F5.3)
      END
C
```

```
      SUBROUTINE CALCUL(INDICE,MODULO,X,Y,Z)
      REAL * 8 FORM1,FORM2,FORM3,FORM4
      COMMON/FORESC/FORM1(5),FORM2(6),FORM3(6),FORM4(7)
      K=INDICE*MODULO
      V=((X+Y)/Z)+FLOAT(K)
      IF(MODULO.LT.0) V=V+0.173
      IF(INDICE.LT.0) V=V-0.178
      IF(MODULO.GE.0.AND.INDICE.GE.0) WRITE(6,FORM1) V
      IF(MODULO.GE.0.AND.INDICE.LT.0) WRITE(6,FORM2) V
      IF(MODULO.LT.0.AND.INDICE.GE.0) WRITE(6,FORM3) V
      IF(MODULO.LT.0.AND.INDICE.LT.0) WRITE(6,FORM4) V
      RETURN
      END
/*
//GO.SYSIN DD *
(1X,' VELOCIDAD NO CORREGIDA',2X,E14.6)
(1X,' VELOCIDAD CORREGIDA POR EXCESO',2X,E14.6)
(1X,' VELOCIDAD CORREGIDA POR DEFECTO',2X,E14.6)
(1X,' CALCULO 2 VELOCIDADES.NO TENERLAS EN CUENTA')
  2  3 .321 .745-.012
/*
//
```

CAP.V,PROGR.13.CASO 3. Los formatos de escritura preparados ingresan al programa principal por tarjetas leídas con los formatos convencionales 100, 101, 101 y 102, a los vectores FORM1, FORM2, FORM3 y FORM4, declarados REAL*8 y de dimensión 5, 6, 6 y 7 respectivamente. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos preparados se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3,FORM4
      DIMENSION FORM1(5),FORM2(6),FORM3(6),FORM4(7)
      READ(5,100) FORM1
      READ(5,101) FORM2
      READ(5,101) FORM3
      READ(5,102) FORM4
      READ(5,103) INDICE,MODULO,X,Y,Z
      CALL CALCUL(FORM1,FORM2,FORM3,FORM4,INDICE,MODULO,X,Y,Z)
      STOP
  100 FORMAT(5A8)
  101 FORMAT(6A8)
  102 FORMAT(7A8)
  103 FORMAT(2I3,3F5.3)
      END
C
      SUBROUTINE CALCUL(FORM1,FORM2,FORM3,FORM4,INDICE,MODULO,X,Y,Z)
      DIMENSION FORM1(5),FORM2(6),FORM3(6),FORM4(7)
      K=INDICE*MODULO
      V=((X+Y)/Z)+FLOAT(K)
      IF(MODULO.LT.0) V=V+0.173
      IF(INDICE.LT.0) V=V-0.178
```

```
      IF(MODULO.GE.0.AND.INDICE.GE.0) WRITE(6,FORM1) V
      IF(MODULO.GE.0.AND.INDICE.LT.0) WRITE(6,FORM2) V
      IF(MODULO.LT.0.AND.INDICE.GE.0) WRITE(6,FORM3) V
      IF(MODULO.LT.0.AND.INDICE.LT.0) WRITE(6,FORM4) V
      RETURN
      END
/*
//GO.SYSIN DD *
(1X,' VELOCIDAD NO CORREGIDA',2X,E14.6)
(1X,' VELOCIDAD CORREGIDA POR EXCESO',2X,E14.6)
(1X,' VELOCIDAD CORREGIDA POR DEFECTO',2X,E14.6)
(1X,' CALCULO 2 VELOCIDADES.NO TENERLAS EN CUENTA')
  2  3 .321 .745-.012
/*
//
```

CAP.V,PROGR.13,CASO 4. Los formatos de escritura preparados están definidos en el programa principal por medio de sentencias DATA de nombres FORM1, FORM2, FORM3 y FORM4, REAL*8 y de dimensión 5, 6, 7 y 7 respectivamente. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. El programa no tiene rutinas a las cuales se transmitan los formatos preparados.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3,FORM4
      DIMENSION FORM1(5),FORM2(6),FORM3(7),FORM4(7)
      DATA FORM1/'(1X,23H ','VELOCIDA','D NO COR','REGIDA,2','X,E14.6)'/
      DATA FORM2/'(1X,31H ','VELOCIDA','D CORREG','IDA POR ','EXCESO,2',
     1  'X,E14.6)'/
      DATA FORM3/'(1X,32H ','VELOCIDA','D CORREG','IDA POR ','DEFECTO,',
     1  '2X,E14.6',')       '/
      DATA FORM4/'(1X,43HC','ALCULO 2',' VELOCID','ADES.NO ','TENERLAS',
     1  ' EN CUEN','TA)     '/
      READ(5,103) INDICE,MODULO,X,Y,Z
      K=INDICE*MODULO
      V=((X+Y/Z)+FLOAT(K)
      IF(MODULO.LT.0) V=V+0.173
      IF(INDICE.LT.0) V=V-0.178
      IF(MODULO.GE.0.AND.INDICE.GE.0) WRITE(6,FORM1) V
      IF(MODULO.GE.0.AND.INDICE.LT.0) WRITE(6,FORM2) V
      IF(MODULO.LT.0.AND.INDICE.GE.0) WRITE(6,FORM3) V
      IF(MODULO.LT.0.AND.INDICE.LT.0) WRITE(6,FORM4) V
      STOP
  103 FORMAT(2I3,3F5.3)
      END
/*
//GO.SYSIN DD *
  2  3 .321 .745-.012
/*
//
```

CAP.V,PROGR.13,CASO 5. Los formatos de escritura preparados están definidos por

medio de sentencias DATA de nombres FORM1, FORM2, FORM3 y FORM4, REAL*8 y de dimensión 5, 6, 7 y 7 respectivamente, incluídas en un BLOCK DATA. En algún momento de la ejecución se elegirá el formato con el cual se va a escribir. Los formatos de escritura preparados se transmiten a la SUBROUTINE CALCUL por medio de un COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,103) INDICE,MODULO,X,Y,Z
      CALL CALCUL(INDICE,MODULO,X,Y,Z)
      STOP
  103 FORMAT(2I3,3F5.3)
      END
C
      BLOCK DATA
      REAL * 8 FORM1,FORM2,FORM3,FORM4
      COMMON/FORESC/FORM1(5),FORM2(6),FORM3(7),FORM4(7)
      DATA FORM1/'(1X,23H ','VELOCIDA','D NO COR','REGIDA,2','X,E14.6)/
      DATA FORM2/'(1X,31H.','VELOCIDA','D CORREG','IDA POR ','EXCESO,2',
     1  'X,E14.6)'/
      DATA FORM3/'(1X,32H ','VELOCIDA','D CORREG','IDA POR ','DEFECTO,',
     1  '2X,E14.6',')       '/
      DATA FORM4/'(1X,43HC','ALCULO 2',' VELOCID','ADES.NO ','TENERLAS',
     1  ' EN CUEN','TA)     '/
      END
C
      SUBROUTINE CALCUL(INDICE,MODULO,X,Y,Z)
      REAL * 8 FORM1,FORM2,FORM3,FORM4
      COMMON/FORESC/FORM1(5),FORM2(6),FORM3(7),FORM4(7)
      K=INDICE*MODULO
      V=((X+Y)/Z)+FLOAT(K)
      IF(MODULO.LT.0) V=V+0.173
      IF(INDICE.LT.0) V=V-0.178
      IF(MODULO.GE.0.AND.INDICE.GE.0) WRITE(6,FORM1) V
      IF(MODULO.GE.0.AND.INDICE.LT.0) WRITE(6,FORM2) V
      IF(MODULO.LT.0.AND.INDICE.GE.0) WRITE(6,FORM3) V
      IF(MODULO.LT.0.AND.INDICE.LT.0) WRITE(6,FORM4) V
      RETURN
      END
/*
//GO.SYSIN DD *
  2  3 .321 .745-.012
/*
//
```

CAP.V,PROGR.13,CASO 6. Los formatos de escritura preparados están definidos en el programa principal por medio de sentencias DATA de nombres FORM1, FORM2, FORM3 y FORM4, REAL*8 y de dimensión 5, 6, 7 y 7 respectivamente. En algún momento de la ejecución se elige el formato con el cual se va a escribir. Los formatos de escritura preparados se transmiten a la SUBROUTINE CALCUL como argumentos en la

sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM1,FORM2,FORM3,FORM4
      DIMENSION FORM1(5),FORM2(6),FORM3(7),FORM4(7)
      DATA FORM1/'(1X,23H ','VELOCIDA','D NO COR','REGIDA,2','X,E14.6)'/
      DATA FORM2/'(1X,31H ','VELOCIDA','D CORREG','IDA POR ','EXCESO,2',
     1  'X,E14.6)'/
      DATA FORM3/'(1X,32H ','VELOCIDA','D CORREG','IDA POR ','DEFECTO,',
     1  '2X,E14.6',')       '/
      DATA FORM4/'(1X,43HC','ALCULO 2',' VELOCID','ADES.NO ','TENERLAS',
     1  ' EN CUEN','TA)     '/
      READ(5,103) INDICE,MODULO,X,Y,Z
      CALL CALCUL(FORM1,FORM2,FORM3,FORM4,INDICE,MODULO,X,Y,Z)
      STOP
  103 FORMAT(2I3,3F5.3)
      END
C
      SUBROUTINE CALCUL(FORM1,FORM2,FORM3,FORM4,INDICE,MODULO,X,Y,Z)
      DIMENSION FORM1(5),FORM2(6),FORM3(7),FORM4(7)
      K=INDICE*MODULO
      V=((X+Y)/Z+FLOAT(K)
      IF(MODULO.LT.0) V=V+0.173
      IF(INDICE.LT.0) V=V-0.178
      IF(MODULO.GE.0.AND.INDICE.GE.0) WRITE(6,FORM1) V
      IF(MODULO.GE.0.AND.INDICE.LT.0) WRITE(6,FORM2) V
      IF(MODULO.LT.0.AND.INDICE.GE.0) WRITE(6,FORM3) V
      IF(MODULO.LT.0.AND.INDICE.LT.0) WRITE(6,FORM4) V
      RETURN
      END
/*
//GO.SYSIN DD *
  2  3 .321 .745-.012
/*
//
```

CAPITULO VI

ESCRITURA DE TITULOS Y LEYENDAS CON FORMATOS VARIABLES QUE SE INTEGRAN DURANTE LA EJECUCION DEL PROGRAMA

PROGRAMA 14.

OBJETO: El programa tiene un formato de escritura incompleto o formato patrón y partes de formato o componentes que se presentan en forma independiente. Según la decisión de una sentencia IF, el componente adecuado se inserta en el formato patrón para completarlo obteniendo así el título que se desea escribir.

CASOS CONSIDERADOS

A) El formato patrón de escritura y los componentes de formato ingresan al programa principal por tarjetas leídas.

CASO 1. El programa no tiene rutinas a las cuales se transmitan el formato patrón de escritura y los componentes de formato.

CASO 2. El formato patrón de escritura y los componentes de formato se transmiten a una rutina de cálculo por medio de una sentencia COMMON.

CASO 3. El formato patrón de escritura y los componentes de formato se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

B) El formato patrón de escritura y los componentes de formato están definidos por medio de una sentencia DATA o bien en el programa principal o en una subrutina BLOCK DATA.

CASO 4. El programa no tiene rutinas a las cuales se transmitan el formato patrón de escritura y los componentes de formato.

CASO 5. El formato patrón de escritura y los componentes de formato están incluídos en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.

CASO 6. El formato patrón de escritura y los componentes de formato se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

El formato patrón de escritura es la tira de caracteres:

(1X,13HEXP. NUM. ,/)

y los componentes que ocuparán, uno por vez, el espacio en blanco, son los siguientes :

5588HCROMO
6278HTECNECIO
8308HRUTÉNIO

El formato patrón ingresa al programa por tarjeta leída con el formato convencional 100, al vector FORM, REAL*4, de dimensión 9. Los componentes ingresan tam-

bién por tarjeta leída, con el formato convencional 101, a los vectores COMP1, COMP2 y COMP3, todos REAL*4 y de dimensión 4.

Cuando la sentencia IF transfiera el control del programa a la marca 1, el DO 5 completará el formato patrón de escritura insertando los cuatro elementos del vector COMP1 de la siguiente manera:

```
FORM(5)=COMP1(1)
FORM(6)=COMP1(2)
FORM(7)=COMP1(3)
FORM(8)=COMP1(4)
```

A continuación imprimirá ese título, realizará un cálculo, escribirá el resultado e irá al STOP. De la misma manera si el control pasara a la marca 2, el DO 6 completará el formato patrón de escritura con los elementos del vector COMP2 y en la marca 3 el DO 7 lo completará con los elementos del vector COMP3.

b) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 4.

El formato patrón de escritura está definido en la sentencia DATA de nombre FORM, REAL*4, de dimensión 9

```
      DATA FORM/'(1X,','13HE','XP. ','NUM.','    ','    ','    ','    ',
     1  '/)  '/
```

Cuando la sentencia IF transfiera el control del programa a la marca 1, el DO 5 completará el formato patrón de escritura insertando los cuatro elementos del vector COMP1 de la siguiente manera:

```
FORM(5)=COMP1(1)
FORM(6)=COMP1(2)
FORM(7)=COMP1(3)
FORM(8)=COMP1(4)
```

A continuación imprimirá ese título, realizará un cálculo, escribirá el resultado e irá al STOP. De la misma manera si el control pasara a la marca 2, el DO 6 completará el formato patrón de escritura con los elementos del vector COMP2 y en la marca 3 el DO 7 lo completará con los elementos del vector COMP3.

CAP.VI,PROGR.14,CASO 1. El formato patrón de escritura ingresa al programa principal por tarjeta leída con el formato convencional 100, al vector de nombre FORM, REAL*4 y de dimensión 9; los componentes de formato ingresan al programa principal por tarjetas leídas con el formato convencional 101, a los vectores COMP1, COMP2 y COMP3, todos REAL*4 y de dimensión 4. En algún momento de la ejecución uno de los componentes se insertará en el formato patrón de escritura completándolo para escribir un título. El programa no tiene rutinas a las cuales se transmitan el formato patrón y los componentes de formato.

```
// .... JOB ........
```

```
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION FORM(9),COMP1(4),COMP2(4),COMP3(4)
      READ(5,100) FORM
      READ(5,101) COMP1
      READ(5,101) COMP2
      READ(5,101) COMP3
      READ(5,102) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 DO 5 I=1,4
    5 FORM(I+4)=COMP1(I)
      WRITE(6,FORM)
      A=B+C
      WRITE(6,103) A
      GO TO 4
    2 DO 6 I=1,4
    6 FORM(I+4)=COMP2(I)
      WRITE(6,FORM)
      A=B-C
      WRITE(6,103) A
      GO TO 4
    3 DO 7 I=1,4
    7 FORM(I+4)=COMP3(I)
      WRITE(6,FORM)
      A=-C-D
      WRITE(6,103) A
    4 STOP
  100 FORMAT(9A4)
  101 FORMAT(4A4)
  102 FORMAT(I3,3F7.2)
  103 FORMAT(1X,' A=',F7.2)
      END
/*
//GO.SYSIN DD *
(1X,13HEXP. NUM.                ,/)
 5588HCROMO
 6278HTECNECIO
 8308HRUTENIO
  0 -14.02  84.61   7.74
/*
//
```

CAP.VI,PROGR.14,CASO 2. El formato patrón de escritura ingresa al programa principal por tarjeta leída con el formato convencional 100, al vector de nombre FORM, REAL*4 y de dimensión 9; los componentes de formato ingresan al programa principal por tarjetas leídas con el formato convencional 101, a los vectores COMP1, COMP2 y COMP3, todos REAL*4 y de dimensión 4. En algún momento de la ejecución uno de los componentes se insertará en el formato patrón de escritura completándolo para escribir un título. El formato patrón y los componentes de formato se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

// .... JOB .....

```
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/FORESC/FORM(9),COMP1(4),COMP2(4),COMP3(4)
      READ(5,100) FORM
      READ(5,101) COMP1
      READ(5,101) COMP2
      READ(5,101) COMP3
      READ(5,102) NUMERO,B,C,D
      CALL CALCUL(NUMERO,B,C,D)
      STOP
  100 FORMAT(9A4)
  101 FORMAT(4A4)
  102 FORMAT(I3,3F7.2)
      END
C
      SUBROUTINE CALCUL(NUMERO,B,C,D)
      COMMON/FORESC/FORM(9),COMP1(4),COMP2(4),COMP3(4)
      IF(NUMERO) 1,2,3
    1 DO 5 I=1,4
    5 FORM(I+4)=COMP1(I)
      WRITE(6,FORM)
      A=B+C
      WRITE(6,103) A
      GO TO 4
    2 DO 6 I=1,4
    6 FORM(I+4)=COMP2(I)
      WRITE(6,FORM)
      A=B-C
      WRITE(6,103) A
      GO TO 4
    3 DO 7 I=1,4
    7 FORM(I+4)=COMP3(I)
      WRITE(6,FORM)
      A=-C-D
      WRITE(6,103) A
    4 RETURN
  103 FORMAT(1X,' A=',F7.2)
      END
/*
//GO.SYSIN DD *
(1X,13HEXP. NUM.                ,/)
 5588HCROMO
 6278HTECNECIO
 8308HRUTENIO
  0 -14.02  84.61   7.74
/*
//
```

CAP.VI,PROGR.14,CASO 3. El formato patrón de escritura ingresa al programa principal por tarjeta leída con el formato convencional 100, al vector de nombre FORM, REAL*4 y de dimensión 9; los componentes de formato ingresan al programa principal por tarjetas leídas con el formato convencional 101, a los vectores COMP1, COMP2 y COMP3, todos REAL*4 y de dimensión 4. En algún momento de la ejecución uno de los componentes se insertará en el formato patrón de escritura com-

pletándolo para escribir un título. El formato patrón y los componentes de formato se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION FORM(9),COMP1(4),COMP2(4),COMP3(4)
      READ(5,100) FORM
      READ(5,101) COMP1
      READ(5,101) COMP2
      READ(5,101) COMP3
      READ(5,102) NUMERO,B,C,D
      CALL CALCUL(FORM,COMP1,COMP2,COMP3,NUMERO,B,C,D)
      STOP
  100 FORMAT(9A4)
  101 FORMAT(4A4)
  102 FORMAT(I3,3F7.2)
      END
C
      SUBROUTINE CALCUL(FORM,COMP1,COMP2,COMP3,NUMERO,B,C,D)
      DIMENSION FORM(9),COMP1(4),COMP2(4),COMP3(4)
      IF(NUMERO) 1,2,3
    1 DO 5 I=1,4
    5 FORM(I+4)=COMP1(I)
      WRITE(6,FORM)
      A=B+C
      WRITE(6,103) A
      GO TO 4
    2 DO 6 I=1,4
    6 FORM(I+4)=COMP2(I)
      WRITE(6,FORM)
      A=B-C
      WRITE(6,103) A
      GO TO 4
    3 DO 7 I=1,4
    7 FORM(I+4)=COMP3(I)
      WRITE(6,FORM)
      A=-C-D
      WRITE(6,103) A
    4 RETURN
  103 FORMAT(1X,' A=',F7.2)
      END
/*
//GO.SYSIN DD *
(1X,13HEXP. NUM.                ,/)
 5588HCROMO
 6278HTECNECIO
 8308HRUTENIO
  0 -14.02  84.61   7.74
/*
//
```

CAP.VI,PROGR.14,CASO 4. El formato patrón de escritura está definido en el pro-

grama principal por medio de una sentencia DATA de nombre FORM, REAL*4 y de dimensión 9; los componentes de formato están definidos por medio de las sentencias DATA de nombres COMP1, COMP2 y COMP3, todas REAL*4 y de dimensión 4. En algún momento de la ejecución uno de los componentes se insertará en el formato patrón de escritura completándolo para escribir un título. El programa no tiene rutinas a las cuales se transmitan el formato patrón y los componentes de formato.

```
// .... JOB .....
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION FORM(9),COMP1(4),COMP2(4),COMP3(4)
      DATA FORM/'(1X,','13HE','XP. ','NUM.','    ','    ','    ','    ',
     1 '/)  '/
      DATA COMP1/' 558','8HCR','OMO ','    '/
      DATA COMP2/' 627','8HTE','CNEC','IO  '/
      DATA COMP3/' 830','8HRU','TENI','O   '/
      READ(5,101) NUMERO,B,C,D
      IF(NUMERO) 1,2,3
    1 DO 5 I=1,4
    5 FORM(I+4)=COMP1(I)
      WRITE(6,FORM)
      A=B+C
      WRITE(6,103) A
      GO TO 4
    2 DO 6 I=1,4
    6 FORM(I+4)=COMP2(I)
      WRITE(6,FORM)
      A=B-C
      WRITE(6,103) A
      GO TO 4
    3 DO 7 I=1,4
    7 FORM(I+4)=COMP3(I)
      WRITE(6,FORM)
      A=-C-D
      WRITE(6,103) A
    4 STOP
  101 FORMAT(I3,3F7.2)
  103 FORMAT(1X,' A=',F7.2)
      END
/*
//GO.SYSIN DD *
  0 -14.02  84.61   7.74
/*
//
```

CAP.VI,PROGR.14,CASO 5. El formato patrón de escritura está definido por medio de una sentencia DATA de nombre FORM, REAL*4 y de dimensión 9 incluída en un BLOCK DATA; los componentes de formato están definidos por medio de las sentencias DATA de nombres COMP1, COMP2 y COMP3, todas REAL*4 y de dimensión 4, incluídas en el mismo BLOCK DATA. En algún momento de la ejecución uno de los componentes se insertará en el formato patrón completándolo para escribir un título. El formato

patrón y los componentes de formato se transmiten a la SUBROUTINE CALCUL por medio de un COMMON.

```
// .... JOB .....
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,101) NUMERO,B,C,D
      CALL CALCUL(NUMERO,B,C,D)
      STOP
  101 FORMAT(I3,3F7.2)
      END
C
      BLOCK DATA
      COMMON/FORESC/FORM(9),COMP1(4),COMP2(4),COMP3(4)
      DATA FORM/'(1X,','13HE','XP. ','NUM.','    ','    ','    ','    ',
     1 '/)  '/
      DATA COMP1/' 558','8HCR','OMO ','    '/
      DATA COMP2/' 627','8HTE','CNEC','IO  '/
      DATA COMP3/' 830','8HRU','TENI','O   '/
      END
C
      SUBROUTINE CALCUL(NUMERO,B,C,D)
      COMMON/FORESC/FORM(9),COMP1(4),COMP2(4),COMP3(4)
      IF(NUMERO) 1,2,3
    1 DO 5 I=1,4
    5 FORM(I+4)=COMP1(I)
      WRITE(6,FORM)
      A=B+C
      WRITE(6,103) A
      GO TO 4
    2 DO 6 I=1,4
    6 FORM(I+4)=COMP2(I)
      WRITE(6,FORM)
      A=B-C
      WRITE(6,103) A
      GO TO 4
    3 DO 7 I=1,4
    7 FORM(I+4)=COMP3(I)
      WRITE(6,FORM)
      A=-C-D
      WRITE(6,103) A
    4 RETURN
  103 FORMAT(1X,' A=',F7.2)
      END
/*
//GO.SYSIN DD *
  0 -14.02  84.61   7.74
/*
//
```

CAP.VI,PROGR.14,CASO 6. El formato patrón de escritura está definido en el programa principal por medio de una sentencia DATA de nombre FORM, REAL*4 y de dimensión 9; los componentes de formato están definidos por medio de las sentencias DATA de nombres COMP1, COMP2 y COMP3, todas REAL*4 y de dimensión 4. En algún mo-

mento de la ejecución uno de los componentes se insertará en el formato patrón de escritura completándolo para escribir un título. El formato patrón y los componentes de formato se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION FORM(9),COMP1(4),COMP2(4),COMP3(4)
      DATA FORM/'(1X,','13HE','XP. ','NUM.','    ','    ','    ','    ',
     1 '/)  '/
      DATA COMP1/' 558','8HCR','OMO ','    '/
      DATA COMP2/' 627','8HTE','CNEC','IO  '/
      DATA COMP3/' 830','8HRU','TENI','O   '/
      READ(5,101) NUMERO,B,C,D
      CALL CALCUL(FORM,COMP1,COMP2,COMP3,NUMERO,B,C,D)
      STOP
  101 FORMAT(I3,3F7.2)
      END
C
      SUBROUTINE CALCUL(FORM,COMP1,COMP2,COMP3,NUMERO,B,C,D)
      DIMENSION FORM(9),COMP1(4),COMP2(4),COMP3(4)
      IF(NUMERO) 1,2,3
    1 DO 5 I=1,4
    5 FORM(I+4)=COMP1(I)
      WRITE(6,FORM)
      A=B+C
      WRITE(6,103) A
      GO TO 4
    2 DO 6 I=1,4
    6 FORM(I+4)=COMP2(I)
      WRITE(6,FORM)
      A=B-C
      WRITE(6,103) A
      GO TO 4
    3 DO 7 I=1,4
    7 FORM(I+4)=COMP3(I)
      WRITE(6,FORM)
      A=-C-D
      WRITE(6,103) A
    4 RETURN
  103 FORMAT(1X,' A=',F7.2)
      END
/*
//GO.SYSIN DD *
  0 -14.02  84.61   7.74
/*
//
```

PROGRAMA 15

OBJETO: El programa tiene un formato de escritura incompleto o formato patrón y dos grupos de componentes de distinta procedencia. Según la decisión de sentencias IF, los componentes adecuados de cada uno de los grupos se insertarán en el formato patrón para completarlo y obtener así la leyenda que se desea escribir.

CASOS CONSIDERADOS

A) El formato patrón de escritura y los componentes de formato de ambos grupos ingresan al programa principal por tarjetas leídas.

CASO 1. El programa no tiene rutinas a las cuales se transmitan el formato patrón de escritura y los componentes de formato de ambos grupos.
CASO 2. El formato patrón de escritura y los componentes de formato de ambos grupos se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. El formato patrón de escritura y los componentes de formato de ambos grupos se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

B) El formato patrón de escritura y los componentes de formato de ambos grupos están definidos por medio de una sentencia DATA, o en el programa principal o en una subrutina BLOCK DATA.

CASO 4. El programa no tiene rutinas a las cuales se transmitan el formato patrón de escritura y los componentes de formato de ambos grupos.
CASO 5. El formato patrón de escritura y los componentes de formato de ambos grupos están incluídos en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. El formato patrón de escritura y los componentes de formato de ambos grupos se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

El formato patrón de escritura es la tira de caracteres
(1X, 2X,F8.6)
Los componentes del primer grupo que ocuparán, uno por vez, la parte anterior del espacio en blanco del formato patrón, son los siguientes:
6HMTD01 6HMTD02 6HMTD03
Los componentes del segundo grupo que ocuparán, uno por vez, la parte posterior

del espacio en blanco, son los siguientes:
6HAJUS1 6HAJUS2 6HAJUS3 6HAJUS4 6HAJUS5 6HAJUS6

El formato patrón ingresa al programa por tarjeta leída con el formato convencional 101, al vector FORM, declarado REAL*8 y de dimensión 4. Los componentes del primer grupo ingresan por tarjeta leída con el formato convencional 102, al vector METODO, declarado REAL*8 y de dimensión 3. Los componentes del segundo grupo ingresan por tarjeta leída con el formato convencional 103, al vector AJUSTE, declarado REAL*8 y de dimensión 6.

Cuando la sentencia IF(NUMERO) 1,2,3 transfiere el control del programa a la marca 1, después de realizar un cálculo, se llenará el formato patrón incompleto con las tiras de caracteres MTDO1 AJUS1 mediante las sentencias

```
FORM(2)=METODO(1)
FORM(3)=AJUSTE(1)
```

y se imprimirá la variable Al junto con la leyenda apropiada. Una nueva sentencia IF decidirá continuar el programa secuencialmente o transferir el control a la marca 5. En el primer caso, después de realizar un cálculo, mediante la sentencia

```
FORM(3)=AJUSTE(2)
```

se rellenará el formato patrón con la tira de caracteres MTDO1 AJUS2, pero si el control pasara a la marca 5, después de realizar un nuevo cálculo se preparará la leyenda con la variante MTDO1 AJUS3 mediante la sentencia

```
FORM(3)=AJUSTE(3)
```

De manera análoga si la sentencia IF(NUMERO)1,2,3 derivara el programa a la marca 2 se prepararían leyendas con las tiras de caracteres

MTDO2 AJUS1
MTDO2 AJUS4
MTDO2 AJUS2

y si el control pasara a la marca 3 se prepararían las leyendas con las tiras de caracteres

MTDO3 AJUS6
MTDO3 AJUS5
MTDO3 AJUS3

b) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 4.

El formato patrón de escritura está definido en la sentencia DATA de nombre FORM, declarada REAL*8 y de dimensión 4

```
DATA FORM/8H  (1X,  ,1H ,1H ,8H2X,F8.6)/
```

en donde la convención ,1H , rellena ese espacio con 8 blancos, es decir, produce el mismo efecto que escribir ,8H ,.

Los componentes del primer grupo que ocuparán, uno por vez, la parte anterior del espacio en blanco del formato patrón están definidos en la sentencia DATA de

nombre METODO, declarada REAL*8 y de dimensión 3

```
      DATA METODO/8H6HMTDO1 ,8H6HMTDO2 ,8H6HMTDO3 /
```

en donde debe distinguirse entre las haches del conjunto 8H, propio del DATA, que abarca 8 caracteres (REAL*8) y está repetido 3 veces (dimensión 3), de las haches del conjunto 6H cuya función es especificar la tira de caracteres que se espera imprimir.

Los componentes del segundo grupo que ocuparán, uno por vez, la parte posterior del espacio en blanco del formato patrón, están definidos en la sentencia DATA de nombre AJUSTE, declarada REAL*8 y de dimensión 6

```
      DATA AJUSTE/8H6HAJUS1 ,8H6HAJUS2 ,8H6HAJUS3 ,8H6HAJUS4 ,
     1 8H6HAJUS5 ,8H6HAJUS6 /
```

Cuando la sentencia IF(NUMERO) 1,2,3 transfiere el control del programa a la marca 1, después de realizar un cálculo, se llenará el formato patrón incompleto con las tiras de caracteres MTDO1 AJUS1 mediante las sentencias

```
      FORM(2)=METODO(1)
      FORM(3)=AJUSTE(1)
```

y se imprimirá la variable A1 junto con la leyenda apropiada. Una nueva sentencia IF decidirá continuar el programa secuencialmente o transferir el control a la marca 5. En el primer caso, después de realizar un cálculo, mediante la sentencia

```
      FORM(3)=AJUSTE(2)
```

se rellenará el formato patrón con la tira de caracteres MTDO1 AJUS2; pero si el control pasara a la marca 5, después de realizar un nuevo cálculo, se preparará la leyenda con la variante MTDO1 AJUS3 mediante la sentencia

```
      FORM(3)=AJUSTE(3)
```

De manera análoga si la sentencia IF(NUMERO) 1,2,3 derivara el programa a la marca 2 se prepararían leyendas con las tiras de caracteres

```
MTDO2 AJUS1
MTDO2 AJUS4
MTDO2 AJUS2
```

y si el control pasara a la marca 3 se prepararían las leyendas con las tiras de caracteres

```
MTDO3 AJUS6
MTDO3 AJUS5
MTDO3 AJUS3
```

CAP.VI,PROGR.15,CASO 1. El formato patrón de escritura ingresa al programa principal por tarjeta leída con el formato convencional 101, al vector de nombre FORM declarado REAL*8 y de dimensión 4. Los componentes del primer grupo ingresan por tarjeta leída con el formato convencional 102, al vector METODO, declarado REAL*8 y de dimensión 3.Los componentes del segundo grupo ingresan por tarjeta leída con el formato convencional 103, al vector AJUSTE, declarado REAL*8 y de di-

mensión 6. En algún momento de la ejecución un componente de cada grupo se insertará en el formato patrón de escritura y lo completará para escribir una leyenda. El programa no tiene rutinas a las cuales se transmitan el formato patrón y los dos grupos de componentes de formato.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM,METODO,AJUSTE
      DIMENSION FORM(4),METODO(3),AJUSTE(6)
      READ(5,101) FORM
      READ(5,102) METODO
      READ(5,103) AJUSTE
      READ(5,100) NUMERO,B,C
      IF(NUMERO) 1,2,3
    1 A=B+C
      A1=A-0.03
      FORM(2)=METODO(1)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.3) GO TO 5
      A2=A-0.04
      FORM(3)=AJUSTE(2)
      WRITE(6,FORM) A2
      GO TO 4
    5 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
      GO TO 4
    2 A=B+C+SQRT(B*C)
      A1=A-0.03
      FORM(2)=METODO(2)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.25) GO TO 6
      A2=A-0.01
      FORM(3)=AJUSTE(4)
      WRITE(6,FORM) A2
      GO TO 4
    6 A2=A+0.015
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A2
      GO TO 4
    3 A=B+C+(0.5*(B+C))
      A1=A+0.015
      FORM(2)=METODO(3)
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A1
      IF(A.LE.0.24) GO TO 7
      A2=A+0.07
      FORM(3)=AJUSTE(5)
      WRITE(6,FORM) A2
      GO TO 4
    7 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
```

```
    4 STOP
  100 FORMAT(I2,2F6.2)
  101 FORMAT(4A8)
  102 FORMAT(3A8)
  103 FORMAT(6A8)
      END
/*
//GO.SYSIN DD *
  (1X,                    2X,F8.6)
6HMTDO1 6HMTDO2 6HMTDO3
6HAJUS1 6HAJUS2 6HAJUS3 6HAJUS4 6HAJUS5 6HAJUS6
-5  1.52  0.73
/*
//
```

CAP.VI,PROGR.15,CASO 2. El formato patrón de escritura ingresa al programa principal por tarjeta leída con el formato convencional 101, al vector de nombre FORM, declarado REAL*8 y de dimensión 4. Los componentes del primer grupo ingresan por tarjeta leída con el formato convencional 102, al vector METODO, declarado REAL*8 y de dimensión 3. Los componentes del segundo grupo ingresan por tarjeta leída con el formato convencional 103, al vector AJUSTE, declarado REAL*8 y de dimensión 6. En algún momento de la ejecución un componente de cada grupo se insertará en el formato patrón de escritura y lo completará para escribir una leyenda. El formato patrón y los dos grupos de componentes de formato se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM,METODO,AJUSTE
      COMMON/FORESC/FORM(4),METODO(3),AJUSTE(6)
      READ(5,101) FORM
      READ(5,102) METODO
      READ(5,103) AJUSTE
      READ(5,100) NUMERO,B,C
      CALL CALCUL(NUMERO,B,C)
      STOP
  100 FORMAT(I2,2F6.2)
  101 FORMAT(4A8)
  102 FORMAT(3A8)
  103 FORMAT(6A8)
      END
C
      SUBROUTINE CALCUL(NUMERO,B,C)
      REAL * 8 FORM,METODO,AJUSTE
      COMMON/FORESC/FORM(4),METODO(3),AJUSTE(6)
      IF(NUMERO) 1,2,3
    1 A=B+C
      A1=A-0.03
      FORM(2)=METODO(1)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
```

```
      IF(A.GT.0.3) GO TO 5
      A2=A-0.04
      FORM(3)=AJUSTE(2)
      WRITE(6,FORM) A2
      GO TO 4
    5 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
      GO TO 4
    2 A=B+C+SQRT(B*C)
      A1=A-0.03
      FORM(2)=METODO(2)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.25) GO TO 6
      A2=A-0.01
      FORM(3)=AJUSTE(4)
      WRITE(6,FORM) A2
      GO TO 4
    6 A2=A+0.015
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A2
      GO TO 4
    3 A=B+C+(0.5*(B+C))
      A1=A+0.015
      FORM(2)=METODO(3)
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A1
      IF(A.LE.0.24) GO TO 7
      A2=A+0.07
      FORM(3)=AJUSTE(5)
      WRITE(6,FORM) A2
      GO TO 4
    7 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
    4 RETURN
      END
/*
//GO.SYSIN DD *
  (1X,                    2X,F8.6)
6HMTDO1 6HMTDO2 6HMTDO3
6HAJUS1 6HAJUS2 6HAJUS3 6HAJUS4 6HAJUS5 6HAJUS6
-5  1.52  0.73
/*
//
```

CAP.VI,PROGR.15,CASO 3. El formato patrón de escritura ingresa al programa principal por tarjeta leída con el formato convencional 101, al vector de nombre FORM, declarado REAL*8 y de dimensión 4. Los componentes del primer grupo ingresan por tarjeta leída con el formato convencional 102, al vector METODO, declarado REAL*8 y de dimensión 3. Los componentes del segundo grupo ingresan por tarjeta leída con el formato convencional 103, al vector AJUSTE, declarado REAL*8 y de

dimensión 6. En algún momento de la ejecución un componente de cada grupo se insertará en el formato patrón de escritura y lo completará para escribir una leyenda. El formato patrón y los dos grupos de componentes de formato se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM,METODO,AJUSTE
      DIMENSION FORM(4),METODO(3),AJUSTE(6)
      READ(5,101) FORM
      READ(5,102) METODO
      READ(5,103) AJUSTE
      READ(5,100) NUMERO,B,C
      CALL CALCUL(FORM,METODO,AJUSTE,NUMERO,B,C)
      STOP
  100 FORMAT(I2,2F6.2)
  101 FORMAT(4A8)
  102 FORMAT(3A8)
  103 FORMAT(6A8)
      END
C
      SUBROUTINE CALCUL(FORM,METODO,AJUSTE,NUMERO,B,C)
      REAL * 8 FORM,METODO,AJUSTE
      DIMENSION FORM(4),METODO(3),AJUSTE(6)
      IF(NUMERO) 1,2,3
    1 A=B+C
      A1=A-0.03
      FORM(2)=METODO(1)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.3) GO TO 5
      A2=A-0.04
      FORM(3)=AJUSTE(2)
      WRITE(6,FORM) A2
      GO TO 4
    5 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
      GO TO 4
    2 A=B+C+SQRT(B*C)
      A1=A-0.03
      FORM(2)=METODO(2)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.25) GO TO 6
      A2=A-0.01
      FORM(3)=AJUSTE(4)
      WRITE(6,FORM) A2
      GO TO 4
    6 A2=A+0.015
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A2
      GO TO 4
    3 A=B+C+(0.5*(B+C))
```

```
      A1=A+0.015
      FORM(2)=METODO(3)
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM)A1
      IF(A.LE.0.24) GO TO 7
      A2=A+0.07
      FORM(3)=AJUSTE(5)
      WRITE(6,FORM) A2
      GO TO 4
    7 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
    4 RETURN
      END
/*
//GO.SYSIN DD *
  (1X,                    2X,F8.6)
6HMTDO1 6HMTDO2 6HMTDO3
6HAJUS1 6HAJUS2 6HAJUS3 6HAJUS4 6HAJUS5 6HAJUS6
-5  1.52  0.73
/*
//
```

CAP.VI,PROGR.15,CASO 4. El formato patrón de escritura está definido en el programa principal por medio de una sentencia DATA de nombre FORM, declarada REAL*8 y de dimensión 4. Los componentes del primer grupo están definidos por medio de una sentencia DATA de nombre METODO, declarada REAL*8 y de dimensión 3. Los componentes del segundo grupo están definidos por medio de una sentencia DATA de nombre AJUSTE, declarada REAL*8 y de dimensión 6. En algún momento de la ejecución un componente de cada grupo se insertará en el formato patrón de escritura y lo completará para escribir una leyenda. El programa no tiene rutinas a las cuales se transmitan el formato patrón y los dos grupos de componentes de formato.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM,METODO,AJUSTE
      DIMENSION FORM(4),METODO(3),AJUSTE(6)
      DATA FORM/8H  (1X,  ,1H ,1H ,8H2X,F8.6)/
      DATA METODO/8H6HMTDO1 ,8H6HMTDO2 ,8H6HMTDO3 /
      DATA AJUSTE/8H6HAJUS1 ,8H6HAJUS2 ,8H6HAJUS3 ,8H6HAJUS4 ,
     1  8H6HAJUS5 ,8H6HAJUS6 /
      READ(5,100) NUMERO,B,C
      IF(NUMERO) 1,2,3
    1 A=B+C
      A1=A-0.03
      FORM(2)=METODO(1)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.3) GO TO 5
      A2=A-0.04
      FORM(3)=AJUSTE(2)
```

```
      WRITE(6,FORM) A2
      GO TO 4
    5 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
      GO TO 4
    2 A=B+C+SQRT(B*C)
      A1=A-0.03
      FORM(2)=METODO(2)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.25) GO TO 6
      A2=A-0.01
      FORM(3)=AJUSTE(4)
      WRITE(6,FORM) A2
      GO TO 4
    6 A2=A+0.015
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A2
      GO TO 4
    3 A=B+C+(0.5*(B+C))
      A1=A+0.015
      FORM(2)=METODO(3)
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A1
      IF(A.LE.0.24) GO TO 7
      A2=A+0.07
      FORM(3)=AJUSTE(5)
      WRITE(6,FORM) A2
      GO TO 4
    7 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
    4 STOP
  100 FORMAT(I2,2F6.2)
      END
/*
//GO.SYSIN DD *
 5  1.52  0.73
/*
//
```

CAP.VI,PROGR.15,CASO 5. El formato patrón de escritura está definido por medio de una sentencia DATA de nombre FORM, declarada REAL*8 y de dimensión 4, incluída en un BLOCK DATA. Los componentes del primer grupo están definidos por medio de una sentencia DATA de nombre METODO, declarada REAL*8 y de dimensión 3, incluída en el mismo BLOCK DATA. Los componentes del segundo grupo están definidos por medio de una sentencia DATA de nombre AJUSTE, declarada REAL*8 y de dimensión 6, incluída también en ese mismo BLOCK DATA. En algún momento de la ejecución un componente de cada grupo se insertará en el formato patrón de escritura y lo completará para escribir una leyenda. El formato patrón y los dos grupos de componentes

de formato se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) NUMERO,B,C
      CALL CALCUL(NUMERO,B,C)
      STOP
  100 FORMAT(I2,2F6.2)
      END
C
      BLOCK DATA
      REAL * 8 FORM,METODO,AJUSTE
      COMMON/FORESC/FORM(4),METODO(3),AJUSTE(6)
      DATA FORM/8H  (1X,  ,1H ,1H ,8H2X,F8.6)/
      DATA METODO/8H6HMTDO1 ,8H6HMTDO2 ,8H6HMTDO3 /
      DATA AJUSTE/8H6HAJUS1 ,8H6HAJUS2 ,8H6HAJUS3 ,8H6HAJUS4 ,
     1  8H6HAJUS5 ,8H6HAJUS6 /
      END
C
      SUBROUTINE CALCUL(NUMERO,B,C)
      REAL * 8 FORM,METODO,AJUSTE
      COMMON/FORESC/FORM(4),METODO(3),AJUSTE(6)
      IF(NUMERO) 1,2,3
    1 A=B+C
      A1=A-0.03
      FORM(2)=METODO(1)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.3) GO TO 5
      A2=A-0.04
      FORM(3)=AJUSTE(2)
      WRITE(6,FORM) A2
      GO TO 4
    5 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
      GO TO 4
    2 A=B+C+SQRT(B*C)
      A1=A-0.03
      FORM(2)=METODO(2)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.25) GO TO 6
      A2=A-0.01
      FORM(3)=AJUSTE(4)
      WRITE(6,FORM) A2
      GO TO 4
    6 A2=A+0.015
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A2
      GO TO 4
    3 A=B+C+(0.5*(B+C))
      A1=A+0.015
      FORM(2)=METODO(3)
```

```
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A1
      IF(A.LE.0.24) GO TO 7
      A2=A+0.07
      FORM(3)=AJUSTE(5)
      WRITE(6,FORM) A2
      GO TO 4
    7 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
    4 RETURN
      END
/*
//GO.SYSIN DD *
 5  1.52  0.73
/*
//
```

CAP.VI,PROGR.15,CASO 6. El formato patrón de escritura está definido en el programa principal por medio de una sentencia DATA de nombre FORM, declarada REAL*8 y de dimensión 4. Los componentes del primer grupo están definidos por medio de una sentencia DATA de nombre METODO , declarada REAL*8 y de dimensión 3. Los componentes del segundo grupo están definidos en una sentencia DATA de nombre AJUSTE, declarada REAL*8 y de dimensión 6. En algún momento de la ejecución un componente de cada grupo se insertará en el formato patrón de escritura y lo completará para escribir una leyenda. El formato patrón y los dos grupos de componentes de formato se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 FORM,METODO,AJUSTE
      DIMENSION FORM(4),METODO(3),AJUSTE(6)
      DATA FORM/8H  (1X,  ,1H ,1H ,8H2X,F8.6)/
      DATA METODO/8H6HMTDO1 ,8H6HMTDO2 ,8H6HMTDO3 /
      DATA AJUSTE/8H6HAJUS1 ,8H6HAJUS2 ,8H6HAJUS3 ,8H6HAJUS4 ,
     1  8H6HAJUS5 ,8H6HAJUS6 /
      READ(5,100) NUMERO,B,C
      CALL CALCUL(FORM,METODO,AJUSTE,NUMERO,B,C)
      STOP
  100 FORMAT(I2,2F6.2)
      END
C
      SUBROUTINE CALCUL(FORM,METODO,AJUSTE,NUMERO,B,C)
      REAL * 8 FORM,METODO,AJUSTE
      DIMENSION FORM(4),METODO(3),AJUSTE(6)
      IF(NUMERO) 1,2,3
    1 A=B+C
      A1=A-0.03
      FORM(2)=METODO(1)
```

```
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.3) GO TO 5
      A2=A-0.04
      FORM(3)=AJUSTE(2)
      WRITE(6,FORM) A2
      GO TO 4
    5 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
      GO TO 4
    2 A=B+C+SQRT(B*C)
      A1=A-0.03
      FORM(2)=METODO(2)
      FORM(3)=AJUSTE(1)
      WRITE(6,FORM) A1
      IF(A.GT.0.25) GO TO 6
      A2=A-0.01
      FORM(3)=AJUSTE(4)
      WRITE(6,FORM) A2
      GO TO 4
    6 A2=A+0.015
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A2
      GO TO 4
    3 A=B+C+(0.5*(B+C))
      A1=A+0.015
      FORM(2)=METODO(3)
      FORM(3)=AJUSTE(6)
      WRITE(6,FORM) A1
      IF(A.LE.0.24) GO TO 7
      A2=A+0.07
      FORM(3)=AJUSTE(5)
      WRITE(6,FORM) A2
      GO TO 4
    7 A2=A+0.06
      FORM(3)=AJUSTE(3)
      WRITE(6,FORM) A2
    4 RETURN
      END
/*
//GO.SYSIN DD *
 5  1.52  0.73
/*
//
```

CAPITULO VII

FORMATOS DE ESCRITURA QUE SE COMPONEN DURANTE LA EJECUCION DEL PROGRAMA A BASE DE ELEMENTOS INDEPENDIENTES

PROGRAMA 16.

OBJETO: El programa dispone de elementos de formato de escritura que combinados durante la ejecución, integrarán un formato completo para imprimir literales, enteros y/o flotantes según las exigencias del momento.

CASOS CONSIDERADOS

A) Los elementos de formato de escritura ingresan al programa principal por tarjeta leída.

CASO 1. El programa no tiene rutinas a las cuales se transmitan los elementos de formato de escritura.
CASO 2. Los elementos de formato de escritura se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 3. Los elementos de formato de escritura se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

B) Los elementos de formato de escritura están definidos por medio de una sentencia DATA o en el programa principal o en una subrutina BLOCK DATA.

CASO 4. El programa no tiene rutinas a las cuales se transmitan los elementos de formato de escritura.
CASO 5. Los elementos de formato de escritura están incluídos en un BLOCK DATA y se transmiten a una rutina de cálculo por medio de una sentencia COMMON.
CASO 6. Los elementos de formato de escritura se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

Los elementos de formato de que dispone el programa, todos REAL*4, son los siguientes:

(1X,	Es el paréntesis izquierdo y el caracter de control de carro mas una coma.
A4	Es el código de conversión A para escribir 4 caracteres.
I4	Es el código de conversión I para escribir un número entero de 4 dígitos.
F7.2	Es el código de conversión F para escribir un número flotante con un campo de 7 y 2 decimales.
,4X,	Es un código para imprimir 4 espacios blancos.
)	Es el paréntesis derecho.

Estos elementos figuran en una sola tarjeta

(1X, A4 I4 F7.2,4X,)

e ingresan a las variables PARENI,CARACT,ENTERO,FLOTAN,ESPACI y PAREND por medio del formato convencional 100. El formato de escritura FORM es un vector vacío, REAL*4 de dimensión 7, el cual durante la ejecución del programa se llenará con los elementos de formato señalados. Así, si la decisión de la sentencia IF(NUMERO) 1,2,3 pasa el control del programa a la marca 1, el conjunto de sentencias

```
FORM(1)=PARENI
FORM(2)=CARACT
FORM(3)=ESPACI
FORM(4)=FLOTAN
FORM(5)=PAREND
```

generará el formato de escritura

(1X, A4 ,4X,F7.2)

con el cual se escribirán las variables PALABR y A.

En forma análoga, en la marca 2 se generará el formato de escritura

(1X, I4 ,4X, A4 ,4X,F7.2)

con el cual se escribirán las variables NUMERO,PALABR y A.

Por último en la marca 3 se generará el formato de escritura

(1X, I4 ,4X,F7.2)

con el cual se escribirán las variables NUMERO y A.

b) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 4.

Los elementos de formato de que dispone el programa, todos REAL*4, están definidos en las siguientes sentencias DATA:

```
DATA PARENI/'(1X,'/
DATA CARACT/' A4 '/
DATA ENTERO/' I4 '/
DATA FLOTAN/'F7.2'/
DATA ESPACI/',4X,'/
DATA PAREND/')   '/
```

El formato de escritura FORM es un vector vacío, REAL*4 de dimensión 7, el cual durante la ejecución del programa se llenará con los elementos de formato señalados. Así, si la decisión de la sentencia IF(NUMERO) 1,2,3 pasa el control del programa a la marca 1, el conjunto de sentencia

```
FORM(1)=PARENI
FORM(2)=CARACT
FORM(3)=ESPACI
FORM(4)=FLOTAN
FORM(5)=PAREND
```

generará el formato de escritura

(1X, A4 ,4X,F7.2)

con el cual se escribirán las variables PALABR y A.

En forma análoga en la marca 2 se generará el formato de escritura

(1X, I4 ,4X, A4 ,4X,F7.2)

con el cual se escribirán las variables NUMERO, PALABR y A.

Por último en la marca 3 se generará el formato de escritura

(1X, I4 ,4X,F7.2)

con el cual se escribirán las variables NUMERO y A.

CAP.VII,PROGR.16,CASO 1. Los elementos de formato de escritura ingresan al programa principal por tarjeta leída con el formato convencional 100, a las variables PARENI,CARACT,ENTERO,FLOTAN,ESPACI y PAREND. En algún momento de la ejecución los elementos, convenientemente combinados, llenarán el vector vacío FORM, REAL*4 de dimensión 7 e integrarán el formato de escritura que se necesita. El programa no tiene rutinas a las cuales se transmitan los elementos de formato.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION FORM(7)
      READ(5,100) PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND
      READ(5,101) PALABR
      DO 5 M=1,3
      READ(5,102) I,J,K,B,C,D
      NUMERO=I+J*K
      IF(NUMERO) 1,2,3
    1 A=SQRT(B+C)
      FORM(1)=PARENI
      FORM(2)=CARACT
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) PALABR,A
      GO TO 5
    2 A=SQRT(B-C)
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=CARACT
      FORM(5)=ESPACI
      FORM(6)=FLOTAN
      FORM(7)=PAREND
      WRITE(6,FORM) NUMERO,PALABR,A
      GO TO 5
    3 A=-C-D
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) NUMERO,A
    5 CONTINUE
      STOP
```

```
  100 FORMAT(6A4)
  101 FORMAT(A4)
  102 FORMAT(3I3,3F7.2)
      END
/*
//GO.SYSIN DD *
(1X, A4  I4 F7.2,4X,)
RAIZ
 -5 -3  2  14.02   4.61   7.74
 10 -5  2  63.28 -51.32  11.18
 15 -3  2 -11.11  77.77  22.22
/*
//
```

CAP.VII,PROGR.16,CASO 2. Los elementos de formato de escritura ingresan al programa principal por tarjeta leída con el formato convencional 100, a las variables PARENI,CARACT,ENTERO,FLOTAN,ESPACI y PAREND. En algún momento de la ejecución los elementos, convenientemente combinados, llenarán el vector vacío FORM, REAL*4 de dimensión 7 e integrarán el formato de escritura que se necesita. Los elementos de formato se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      COMMON/FORESC/PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND
      READ(5,100) PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND
      CALL CALCUL
      STOP
  100 FORMAT(6A4)
      END
C
      SUBROUTINE CALCUL
      DIMENSION FORM(7)
      COMMON/FORESC/PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND
      READ(5,101) PALABR
      DO 5 M=1,3
      READ(5,102) I,J,K,B,C,D
      NUMERO=I+J*K
      IF(NUMERO) 1,2,3
    1 A=SQRT(B+C)
      FORM(1)=PARENI
      FORM(2)=CARACT
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) PALABR,A
      GO TO 5
    2 A=SQRT(B-C)
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=CARACT
```

```
      FORM(5)=ESPACI
      FORM(6)=FLOTAN
      FORM(7)=PAREND
      WRITE(6,FORM) NUMERO,PALABR,A
      GO TO 5
    3 A=-C-D
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) NUMERO,A
    5 CONTINUE
      RETURN
  101 FORMAT(A4)
  102 FORMAT(3I3,3F7.2)
      END
/*
//GO.SYSIN DD *
(1X, A4  I4 F7.2,4X,)
RAIZ
 -5 -3  2  14.02   4.61   7.74
 10 -5  2  63.28 -51.32  11.18
 15 -3  2 -11.11  77.77  22.22
/*
//
```

CAP.VII,PROGR.16,CASO 3. Los elementos de formato de escritura ingresan al programa principal por tarjeta leída con el formato convencional 100, a las variables PARENI,CARACT,ENTERO,FLOTAN,ESPACI y PAREND. En algún momento de la ejecución los elementos, convenientemente combinados, llenarán el vector vacío FORM, REAL*4 de dimensión 7 e integrarán el formato de escritura que se necesita. Los elementos de formato se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      READ(5,100) PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND
      CALL CALCUL(PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND)
      STOP
  100 FORMAT(6A4)
      END
C
      SUBROUTINE CALCUL(PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND)
      DIMENSION FORM(7)
      READ(5,101) PALABR
      DO 5 M=1,3
      READ(5,102) I,J,K,B,C,D
      NUMERO=I+J*K
      IF(NUMERO) 1,2,3
    1 A=SQRT(B+C)
      FORM(1)=PARENI
```

```
      FORM(2)=CARACT
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) PALABR,A
      GO TO 5
    2 A=SQRT(B-C)
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=CARACT
      FORM(5)=ESPACI
      FORM(6)=FLOTAN
      FORM(7)=PAREND
      WRITE(6,FORM) NUMERO,PALABR,A
      GO TO 5
    3 A=-C-D
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) NUMERO,A
    5 CONTINUE
      RETURN
  101 FORMAT(A4)
  102 FORMAT(3I3,3F7.2)
      END
/*
//GO.SYSIN DD *
(1X, A4  I4 F7.2,4X,)
RAIZ
 -5 -3  2  14.02   4.61   7.74
 10 -5  2  63.28 -51.32  11.18
 15 -3  2 -11.11  77.77  22.22
/*
//
```

CAP.VII,PROGR.16,CASO 4. Los elementos de formato de escritura están definidos en el programa principal por medio de sentencias DATA de nombres PARENI,CARACT, ENTERO,FLOTAN,ESPACI y PAREND, todas REAL*4. En algún momento de la ejecución los elementos, convenientemente combinados, llenarán el vector vacío FORM, REAL*4 de dimensión 7 e integrarán el formato de escritura que se necesita. El programa no tiene rutinas a las cuales se transmitan los elementos de formato.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DIMENSION FORM(7)
      DATA PARENI/'(1X,'/
      DATA CARACT/' A4 '/
      DATA ENTERO/' I4 '/
      DATA FLOTAN/'F7.2'/
      DATA ESPACI/',4X,'/
```

```
      DATA PAREND/')   '/
      READ(5,101) PALABR
      DO 5 M=1,3
      READ(5,102) I,J,K,B,C,D
      NUMERO=I+J*K
      IF(NUMERO) 1,2,3
    1 A=SQRT(B+C)
      FORM(1)=PARENI
      FORM(2)=CARACT
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) PALABR,A
      GO TO 5
    2 A=SQRT(B-C)
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=CARACT
      FORM(5)=ESPACI
      FORM(6)=FLOTAN
      FORM(7)=PAREND
      WRITE(6,FORM) NUMERO,PALABR,A
      GO TO 5
    3 A=-C-D
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) NUMERO,A
    5 CONTINUE
      STOP
  101 FORMAT(A4)
  102 FORMAT(3I3,3F7.2)
      END
/*
//GO.SYSIN DD *
RAIZ
 -5 -3  2  14.02   4.61   7.74
 10 -5  2  63.28 -51.32  11.18
 15 -3  2 -11.11  77.77  22.22
/*
//
```

CAP.VII,PROGR.16,CASO 5. Los elementos de formato de escritura están definidos por medio de sentencias DATA de nombres PARENI,CARACT,ENTERO,FLOTAN,ESPACI y PAREND, todas REAL*4, incluídas en un BLOCK DATA. En algún momento de la ejecución los elementos, convenientemente combinados, llenarán el vector vacío FORM, REAL*4 de dimensión 7 e integrarán el formato de escritura que se necesita. Los elementos de formato se transmiten a la SUBROUTINE CALCUL por medio de un COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      CALL CALCUL
      STOP
      END
C
      BLOCK DATA
      COMMON/FORESC/PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND
      DATA PARENI/'(1X,'/
      DATA CARACT/' A4 '/
      DATA ENTERO/' I4 '/
      DATA FLOTAN/'F7.2'/
      DATA ESPACI/',4X,'/
      DATA PAREND/')   '/
      END
C
      SUBROUTINE CALCUL
      DIMENSION FORM(7)
      COMMON/FORESC/PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND
      READ(5,101) PALABR
      DO 5 M=1,3
      READ(5,102) I,J,K,B,C,D
      NUMERO=I+J*K
      IF(NUMERO) 1,2,3
    1 A=SQRT(B+C)
      FORM(1)=PARENI
      FORM(2)=CARACT
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) PALABR,A
      GO TO 5
    2 A=SQRT(B-C)
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=CARACT
      FORM(5)=ESPACI
      FORM(6)=FLOTAN
      FORM(7)=PAREND
      WRITE(6,FORM) NUMERO,PALABR,A
      GO TO 5
    3 A=-C-D
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) NUMERO,A
    5 CONTINUE
      RETURN
  101 FORMAT(A4)
  102 FORMAT(3I3,3F7.2)
      END
/*
//GO.SYSIN DD *
```

```
RAIZ
 -5 -3  2  14.02   4.61   7.74
 10 -5  2  63.28 -51.32  11.18
 15 -3  2 -11.11  77.77  22.22
/*
//
```

CAP.VII,PROGR.16,CASO 6. Los elementos de formato de escritura están definidos en el programa principal por medio de sentencias DATA de nombres PARENI,CARACT, ENTERO,FLOTAN,ESPACI y PAREND, todas REAL*4. En algún momento de la ejecución los elementos, convenientemente combinados, llenarán el vector vacío FORM, REAL*4 de dimensión 7 e integrarán el formato de escritura que se necesita. Los elementos de formato se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// ....! JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA PARENI/'(1X,'/
      DATA CARACT/' A4 '/
      DATA ENTERO/' I4 '/
      DATA FLOTAN/'F7.2'/
      DATA ESPACI/',4X,'/
      DATA PAREND/')   '/
      CALL CALCUL(PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND)
      STOP
      END
C
      SUBROUTINE CALCUL(PARENI,CARACT,ENTERO,FLOTAN,ESPACI,PAREND)
      DIMENSION FORM(7)
      READ(5,101) PALABR
      DO 5 M=1,3
      READ(5,102) I,J,K,B,C,D
      NUMERO=I+J*K
      IF(NUMERO) 1,2,3
    1 A=SQRT(B+C)
      FORM(1)=PARENI
      FORM(2)=CARACT
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) PALABR,A
      GO TO 5
    2 A=SQRT(B-C)
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=CARACT
      FORM(5)=ESPACI
      FORM(6)=FLOTAN
      FORM(7)=PAREND
      WRITE(6,FORM) NUMERO,PALABR,A
```

```
      GO TO 5
    3 A=-C-D
      FORM(1)=PARENI
      FORM(2)=ENTERO
      FORM(3)=ESPACI
      FORM(4)=FLOTAN
      FORM(5)=PAREND
      WRITE(6,FORM) NUMERO,A
    5 CONTINUE
      RETURN
  101 FORMAT(A4)
  102 FORMAT(3I3,3F7.2)
      END
/*
//GO.SYSIN DD *
RAIZ
 -5 -3  2  14.02   4.61   7.74
 10 -5  2  63.28 -51.32  11.18
 15 -3  2 -11.11  77.77  22.22
/*
//
```

CAPITULO VIII

CONVERSION DE TIRAS DE CARACTERES A NUMEROS ENTEROS Y FLOTANTES MEDIANTE EL USO DE LA RUTINA ASSEMBLER ZA03AS DE LA "HARWELL SUBROUTINE LIBRARY"

PROGRAMA 17.

OBJETO: El programa tiene como datos tablas que constan de un encabezamiento y un título seguidos de una cierta cantidad, desconocida, de renglones con números. Los datos se leen como tiras de caracteres y después se extraerá la información numérica necesaria convirtiendo los literales en enteros o flotantes según el caso, mediante el uso de la rutina ASSEMBLER ZA03AS de la "Harwell Subroutine Library".

CASOS CONSIDERADOS

A) Las tablas que constituyen los datos ingresan al programa principal por medio de un formato de lectura.

CASO 1. El programa no tiene rutinas a las cuales se transmitan las tablas que constituyen los datos.

CASO 2. Las tablas que constituyen los datos se transmiten a una rutina de cálculo por medio de una sentencia COMMON.

CASO 3. Las tablas que constituyen los datos se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

Los datos del programa son del siguiente tipo:

```
MECA -35
TEMA          CARR          LAMD
 8.53         -6.60         -149
-10.09         8.53          246
-20.31        -19.48         -843
 21.52         26.03         -125
MECA  12
.....................................
.....................................
.....................................
```

en donde cada renglón representa una tarjeta. Siguen otras tablas con encabezamientos y títulos análogos y un número de renglones variable para cada una que impide leer los datos con sentencias DO.

El problema se puede resolver leyendo todos los renglones como caracteres. La rutina ASSEMBLER ZA03AS de la "Harwell Subroutine Library" se encargará de convertir la tira -35 del renglón MECA y todas las tiras de la columna LAMD en números enteros y los caracteres de las columnas TEMA y CARR en números flotantes.

En la marca 1 se leen tarjetas de datos hasta que la sentencia delimitadora /* de por terminado el programa. La tira de caracteres leída ocupa los elementos del

vector TABLA declarado INTEGER*2 y de dimensión 20.

A continuación la primera sentencia IF establece lo siguiente: si la tira empieza por los caracteres "ME" irá a la marca 2 en donde llamará a la rutina ASSEMBLER con argumento (TABLA,40). El número 40 resulta de multiplicar la dimensión de TABLA(20) por los 2 octetos de la declaración INTEGER*2. Estos 40 octetos representan el espacio que la rutina proveerá en una memoria auxiliar en la cual alojará automáticamente el renglón que se acaba de leer.

La sentencia READ(5,101) MECANC que sigue al CALL leerá directamente de la memoria auxiliar los caracteres almacenados en el vector TABLA, pero procesados de acuerdo con el formato FORTRAN utilizado, es decir, la variable MECANC será un número entero. Luego se imprimirá el renglón, se realizará un cálculo para probar que el número obtenido es operable, se escribirá el resultado y el control del programa pasará a la marca 1 a leer otra tarjeta.

La segunda sentencia IF decide que si el renglón empieza por los caracteres "TE", el programa salte a la marca 3 en donde simplemente escribirá la tira, ya que ésta no contiene información numérica e irá a la marca 1 a leer otra tarjeta.

Si el renglón leído no empieza ni por los caracteres "ME" ni por "TE", entonces irá a la marca 4 en donde llamará a la rutina ASSEMBLER nuevamente. La sentencia

```
READ(5,103) TENS,CARGA,LAMBDA
```

que sigue al CALL leerá directamente de la memoria auxiliar los caracteres, pero procesados de acuerdo con el formato FORTRAN utilizado, es decir, las variables TENS y CARGA serán números flotantes y la variable LAMBDA un entero. Luego se imprimirá el renglón, se realizará un cálculo para probar que los números obtenidos son operables, se escribirá el resultado y el control del programa pasará a la marca 1 a leer otra tarjeta. Y así sucesivamente con las demás tablas.

Nótese que las partículas "ME" y "TE" están definidas por medio de sendas sentencias DATA de nombres MECANO e ITECIL, declaradas también INTEGER *2 a los efectos de poder realizar la comparación con el vector TABLA.

CAP.VIII,PROGR.17,CASO 1. Los datos del programa son 3 tablas que se leen íntegramente como caracteres e ingresan al vector TABLA declarado INTEGER*2 y de dimensión 20. Las partículas "ME" y "TE" están definidas por medio de sentencias DATA de nombres MECANO e ITECIL respectivamente, declaradas INTEGER*2 a los efectos de poder compararlas con el vector TABLA. La conversión de caracteres a números enteros y flotantes se realiza con el auxilio de la rutina ASSEMBLER ZA03AS de la "Harwell Subroutine Library". El programa no tiene rutinas a las cuales se transmitan las tablas.

```
// .... JOB ........
```

```
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      INTEGER * 2 TABLA,MECANO,ITECIL
      DIMENSION TABLA(20)
      DATA MECANO/'ME'/
      DATA ITECIL/'TE'/
    1 READ(5,100,END=10) TABLA
      IF(TABLA(4).EQ.MECANO) GO TO 2
      IF(TABLA(4).EQ.ITECIL) GO TO 3
      GO TO 4
    2 CALL ZA03AS(TABLA,40)
      READ(5,101) MECANC
      WRITE(6,102) TABLA
      IFLEXI=MECANC*83
      WRITE(6,104) IFLEXI
      GO TO 1
    3 WRITE(6,102) TABLA
      GO TO 1
    4 CALL ZA03AS(TABLA,40)
      READ(5,103) TENS,CARGA,LAMBDA
      WRITE(6,102) TABLA
      IKA=LAMBDA+MECANC
      RENTA=CARGA*FLOAT(IKA)/TENS
      WRITE(6,105) RENTA
      GO TO 1
   10 STOP
  100 FORMAT(20A2)
  101 FORMAT(10X,I4)
  102 FORMAT(1X,20A2)
  103 FORMAT(6X,F6.2,8X,F6.2,10X,I4)
  104 FORMAT(1H+,50X,'IFLEXI=',I8)
  105 FORMAT(1H+,50X,'RENTA=',E14.6)
      END
/*
//LKED.OTRALIB DD DSN=L10000.HARWELL,DISP=SHR
//LKED.SYSIN DD *,DCB=BLKSIZE=80
         INCLUDE OTRALIB(ZA03AS)
/*
//GO.SYSIN DD *
      MECA -35
      TEMA          CARR          LAMD
        8.53         -6.60          -149
      -10.09          8.53           246
      -20.31        -19.48          -843
       21.52         26.03          -125
      MECA  12
      TEMA          CARR          LAMD
       25.58        -50.16           451
       35.44        -41.12           730
      MECA  28
      TEMA          CARR          LAMD
       60.17         59.46            99
      -65.43         60.35           660
       70.46        -70.48          -123
/*
//
```

CAP.VIII,PROGR.17,CASO 2. Los datos del programa son 3 tablas que se leen íntegramente como caracteres e ingresan al vector TABLA declarado INTEGER*2 y de dimensión 20. Las partículas "ME" y "TE" están definidas por medio de sentencias DATA de nombres MECANO e ITECIL respectivamente, declaradas INTEGER*2 a los efectos de poder compararlas con el vector TABLA. La conversión de caracteres a números enteros y flotantes se realiza con el auxilio de la rutina ASSEMBLER ZA03AS de la "Harwell Subroutine Library". Las tablas se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      INTEGER * 2 TABLA
      COMMON/CARAC/TABLA(20)
    1 READ(5,100,END=10) TABLA
      CALL CALCUL
      GO TO 1
   10 STOP
  100 FORMAT(20A2)
      END
C
      SUBROUTINE CALCUL
      INTEGER * 2 TABLA,MECANO,ITECIL
      COMMON/CARAC/ TABLA(20)
      DATA MECANO/'ME'/
      DATA ITECIL/'TE'/
      IF(TABLA(4).EQ.MECANO) GO TO 2
      IF(TABLA(4).EQ.ITECIL) GO TO 3
      GO TO 4
    2 CALL ZA03AS(TABLA,40)
      READ(5,101)MECANC
      WRITE(6,102) TABLA
      IFLEXI=MECANC*83
      WRITE(6,104) IFLEXI
      RETURN
    3 WRITE(6,102) TABLA
      RETURN
    4 CALL ZA03AS(TABLA,40)
      READ(5,103) TENS,CARGA,LAMBDA
      WRITE(6,102) TABLA
      IKA=LAMBDA+MECANC
      RENTA=CARGA*FLOAT(IKA)/TENS
      WRITE(6,105) RENTA
      RETURN
  101 FORMAT(10X,I4)
  102 FORMAT(1X,20A2)
  103 FORMAT(6X,F6.2,8X,F6.2,10X,I4)
  104 FORMAT(1H+,50X,'IFLEXI=',I8)
  105 FORMAT(1H+,50X,'RENTA=',E14.6)
      END
/*
//LKED.OTRALIB DD DSN=L10000.HARWELL,DISP=SHR
//LKED.SYSIN DD *,DCB=BLKSIZE=80
```

```
          INCLUDE OTRALIB(ZA03AS)
/*
//GO.SYSIN DD *
      MECA -35
      TEMA            CARR            LAMD
        8.53          -6.60           -149
      -10.09           8.53            246
      -20.31         -19.48           -843
       21.52          26.03           -125
      MECA  12
      TEMA            CARR            LAMD
       25.58         -50.16            451
       35.44         -41.12            730
      MECA  28
      TEMA            CARR            LAMD
       60.17          59.46             99
      -65.43          60.35            660
       70.46         -70.48           -123
/*
//
```

CAP.VIII,PROGR.17,CASO 3. Los datos del programa son 3 tablas que se leen íntegramente como caracteres e ingresan al vector TABLA declarado INTEGER*2 y de dimensión 20. Las partículas "ME" y "TE" están definidas por medio de sentencias DATA de nombres MECANO e ITECIL respectivamente, declaradas INTEGER*2 a los efectos de poder compararlas con el vector TABLA. La conversión de caracteres a números enteros y flotantes se realiza con el auxilio de la rutina ASSEMBLER ZA03AS de la "Harwell Subroutine Library". Las tablas se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      INTEGER * 2 TABLA
      DIMENSION TABLA(20)
    1 READ(5,100,END=10) TABLA
      CALL CALCUL(TABLA)
      GO TO 1
   10 STOP
  100 FORMAT(20A2)
      END
C
      SUBROUTINE CALCUL(TABLA)
      INTEGER * 2 TABLA,MECANO,ITECIL
      DIMENSION TABLA(20)
      DATA MECANO/'ME'/
      DATA ITECIL/'TE'/
      IF(TABLA(4).EQ.MECANO) GO TO 2
      IF(TABLA(4).EQ.ITECIL) GO TO 3
      GO TO 4
    2 CALL ZA03AS(TABLA,40)
      READ(5,101) MECANC
```

```
      WRITE(6,102) TABLA
      IFLEXI=MECANC*83
      WRITE(6,104) IFLEXI
      RETURN
    3 WRITE(6,102) TABLA
      RETURN
    4 CALL ZA03AS(TABLA,40)
      READ(5,103) TENS,CARGA,LAMBDA
      WRITE(6,102) TABLA
      IKA=LAMBDA+MECANC
      RENTA=CARGA*FLOAT(IKA)/TENS
      WRITE(6,105) RENTA
      RETURN
  101 FORMAT(10X,I4)
  102 FORMAT(1X,20A2)
  103 FORMAT(6X,F6.2,8X,F6.2,10X,I4)
  104 FORMAT(1H+,50X,'IFLEXI=',I8)
  105 FORMAT(1H+,50X,'RENTA=',E14.6)
      END
/*
//LKED.OTRALIB DD DSN=L10000.HARWELL,DISP=SHR
//LKED.SYSIN DD *,DCB=BLKSIZE=80
         INCLUDE OTRALIB(ZA03AS)
/*
//GO.SYSIN DD *
      MECA -35
      TEMA           CARR           LAMD
        8.53          -6.60          -149
      -10.09           8.53           246
      -20.31         -19.48          -843
       21.52          26.03          -125
      MECA  12
      TEMA           CARR           LAMD
       25.58         -50.16           451
       35.44         -41.12           730
      MECA  28
      TEMA           CARR           LAMD
       60.17          59.46            99
      -65.43          60.35           660
       70.46         -70.48          -123
/*
//
```

PROGRAMA 18.

OBJETO: Los datos del programa son expresiones mixtas que constan de una letra, el signo igual y un número real. Estos datos se leen como si fuesen literales y después se extraerá la información numérica convirtiendo en flotantes las tiras de caracteres que correspondan, mediante el uso de la rutina ASSEMBLER ZA03AS de la "Harwell Subroutine Library".

CASOS CONSIDERADOS

A) Las expresiones mixtas letra-signo igual-número real que constituyen los datos ingresan al programa principal por medio de un formato de lectura.

CASO 1. El programa no tiene rutinas a las cuales se transmitan las expresiones mixtas que constituyen los datos.

CASO 2. Las expresiones mixtas que constituyen los datos se transmiten a una rutina de cálculo por medio de una sentencia COMMON.

CASO 3. Las expresiones mixtas que constituyen los datos se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

Los datos del programa son los siguientes:

```
   A=123456E 03       T=0.3201      G= 541.00574       C=400125.44
R=745.              U=.0                                V=120.87  X=-4571953E 04
                  E=1.35                              F=8547230.1       P=22.63214
      Q=000251.26       W= +9258147.          S=11.98752   H=875421.23
    D=   -78.254   B=  12.524 Y=       8.125874    Z=   -4.23547  O=-3E-02
```

en donde cada renglón representa una tarjeta que se leerá íntegramente como caracteres (incluído el signo igual). La rutina ASSEMBLER ZA03AS de la "Harwell Subroutine Library" se encargará de convertir los caracteres que correspondan en números flotantes.

En la marca 1 se leerán tarjetas de datos hasta que la sentencia delimitadora /* transfiera el control del programa a la marca 9 para realizar pruebas numéricas. La tira de caracteres leída ocupará los elementos del vector TIRA declarado INTEGER*2 y de dimensión 80.

El bucle 8 comprendido entre la marca 8 y la sentencia GO TO 8 leerá todos los datos de cada tarjeta. El DO 2 buscará el signo = del dato; si no lo encontrara (porque todos los datos de esa tarjeta ya fueron procesados) irá a la marca 1 a leer la próxima tarjeta. Si lo encontrara irá a la marca 3: allí establecerá que en K1 empieza el número y que la letra está en KLETRA.

En el DO 7 y a partir de la posición K1 buscará blancos que pudiera haber entre el igual y el primer carácter significativo del número (signo+,signo-,punto o dígito). Una vez rechazados todos los blancos irá a la marca 10 en donde establecerá que el primer carácter válido del número está en la posición K2.

El DO 4 buscará el blanco que siempre tendrá que existir al final del número (inclusive detrás del último dato de una tarjeta); cuando lo encuentre irá a la marca 5 y establecerá que el último dígito del número está en K3. En ese punto del programa llamará a la rutina ASSEMBLER con argumento (INTER,20) en donde INTER es un vector declarado INTEGER*2 y de dimensión 10 porque se supone que el número consta como máximo de 10 caracteres. El número 20 resulta de multiplicar la dimensión de INTER(10) por los 2 octetos de la declaración INTEGER*2; estos 20 octetos representan el espacio que la rutina proveerá en una memoria auxiliar.

A continuación la sentencia

```
WRITE(6,101) (TIRA(L),L=K2,K3)
```

escribirá en dicha memoria los caracteres, pero procesados de acuerdo con el formato FORTRAN utilizado y solamente el número propiamente dicho que está comprendido entre TIRA(K2) y TIRA(K3), es decir, un flotante sin la letra, sin el signo = y sin blancos, con su propio signo en caso que lo tuviera.

Un nuevo llamado a la rutina ASSEMBLER permitirá leer de la memoria auxiliar, con la sentencia READ(5,102) CIFRA, el número flotante con el formato FORTRAN utilizado en dicha sentencia.

El DO 13 buscará, por comparación con el DATA LETRA cuál es la letra del alfabeto que está en el elemento TIRA(KLETRA) o sea, cuál es la letra que se corresponde con el número ya convertido a flotante; cuando la encuentre irá a la marca 12 en donde merced a la sentencia EQUIVALENCE que figura al principio del programa, se establecerá la igualdad entre el número y su letra.

Se corregirá el índice MIN para que busque el dato siguiente de esa misma tarjeta y se volverá a la cabeza del bucle 8.

Cuando todas las tarjetas de datos hayan sido procesadas, el control del programa se transferirá a la marca 9. Allí se realizará una prueba de cálculo que demuestre que todas las letras han sido asignadas a sus correspondientes números y que estos han sido convertidos de manera correcta.

A los efectos de obtener 8 cifras significativas exactas el programa deberá ser ejecutado en doble precisión. Si se pretenden solamente hasta 6 cifras significativas iguales a las de los datos originales, entonces se podrá ejecutar el programa en simple precisión.

Toda vez que se realizare una comparación de tiras de caracteres en una sentencia IF, las variables, constantes o vectores involucrados en la comparación deberán ser del mismo tipo; por esta razón han sido explícitamente declarados

INTEGER*2.

CAP.VIII,PROGR.18,CASO 1. Los datos del programa son expresiones mixtas letra-signo igual- número real, que ingresan al vector TIRA declarado INTEGER*2 y de dimensión 80. El juego de datos consta de cualquier cantidad de tarjetas y cada tarjeta de cualquier cantidad de números dispuetos en cualquier orden respecto de las letras que los acompañan y perforados con cualquier formato de los códigos E o F. Puede haber blancos o ceros entre el signo igual y el primer carácter significativo del número(signo+,signo-,punto o dígito), pero no entre los signos aritméticos +/- y el primer dígito. Cada número tiene que terminar seguido de un blanco, inclusive el último dato de una tarjeta. El programa no tiene rutinas a las cuales se transmitan las expresiones mixtas que constituyen los datos.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      IMPLICIT REAL * 8 (A-H,O-Z)
      INTEGER*2 TIRA(80),INTER(10),BLANCO,IGUAL,LETRA(20)
      DIMENSION ALETRA(20)
      EQUIVALENCE
     1  (ALETRA( 1),A),(ALETRA( 2),B),(ALETRA( 3),C),(ALETRA( 4),D),
     2  (ALETRA( 5),E),(ALETRA( 6),F),(ALETRA( 7),G),(ALETRA( 8),H),
     3  (ALETRA( 9),O),(ALETRA(10),P),(ALETRA(11),Q),(ALETRA(12),R),
     4  (ALETRA(13),S),(ALETRA(14),T),(ALETRA(15),U),(ALETRA(16),V),
     5  (ALETRA(17),W),(ALETRA(18),X),(ALETRA(19),Y),(ALETRA(20),Z)
      DATA IGUAL/'='/,BLANCO/' '/
      DATA LETRA/'A','B','C','D','E','F','G','H','O','P','Q','R','S',
     1  'T','U','V','W','X','Y','Z'/
    1 READ(5,100,END=9) TIRA
      MIN=1
    8 DO 2 I=MIN,80
      IF(TIRA(I).EQ.IGUAL) GO TO 3
    2 CONTINUE
      GO TO 1
    3 K1=I+1
      KLETRA=I-1
      DO 7 I=K1,80
      IF(TIRA(I).NE.BLANCO) GO TO 10
    7 CONTINUE
   10 K2=I
      DO 4 J=K2,80
      IF(TIRA(J).EQ.BLANCO) GO TO 5
    4 CONTINUE
      GO TO 15
    5 K3=J-1
      CALL ZA03AS(INTER,20)
      WRITE(6,101) (TIRA(L),L=K2,K3)
      CALL ZA03AS(INTER,20)
      READ(5,102) CIFRA
      DO 13 IFOR=1,20
      IF(TIRA(KLETRA).EQ.LETRA(IFOR)) GO TO 12
```

```
   13 CONTINUE
   12 ALETRA(IFOR)=CIFRA
      WRITE(6,104) ALETRA(IFOR)
      MIN=K3+1
      GO TO 8
    9 PRUB1=(B+O-T)*(C-X)
      PRUB2=DSQRT(DABS(A+D-V))/(F*S)
      PRUB3=(E/H)*(P-G-U)
      PRUB4=(Q-2.*R)/(W+Y-Z)
      WRITE(6,103) PRUB1,PRUB2,PRUB3,PRUB4
   15 STOP
  100 FORMAT(80A1)
  101 FORMAT(16A1)
  102 FORMAT(D16.8)
  103 FORMAT(1X,'PRUEBA1=',D16.8,2X,'PRUEBA2=',D16.8,2X,'PRUEBA3=',
     1  D16.8,2X,'PRUEBA4=',D16.8)
  104 FORMAT(1X,D16.8)
      END
/*
//LKED.OTRALIB DD DSN=L10000.HARWELL,DISP=SHR
//LKED.SYSIN DD *,DCB=BLKSIZE=80
         INCLUDE OTRALIB(ZA03AS)
/*
//GO.SYSIN DD *
    A=123456E 03       T=0.3201      G= 541.00574       C=400125.44
R=745.              U=.0                                V=120.87  X=-4571953E 04
                   E=1.35                             F=8547230.1       P=22.63214
       Q=000251.26       W= +9258147.         S=11.98752   H=875421.23
     D=   -78.254   B=  12.524 Y=      8.125874          Z=   -4.23547 O=-3E-02
/*
//
```

CAP.VIII,PROGR.18,CASO 2. Los datos del programa son expresiones mixtas letra-signo igual-número real, que ingresan al vector TIRA declarado INTEGER*2 y de dimensión 80. El juego de datos consta de cualquier cantidad de tarjetas y cada tarjeta de cualquier cantidad de números dispuestos en cualquier orden respecto de las letras que los acompañan y perforados con cualquier formato de los códigos E o F. Puede haber blancos o ceros entre el signo igual y el primer carácter significativo del número (signo +,signo-,punto o dígito), pero no entre los signos aritméticos +/- y el primer dígito. Cada número tiene que terminar seguido de un blanco, inclusive el último dato de una tarjeta. Las expresiones mixtas que constituyen los datos se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      IMPLICIT REAL * 8 (A-H,O-Z)
      INTEGER * 2 TIRA
      COMMON/CARAC/TIRA(80),ALETRA(20)
      EQUIVALENCE
```

```
     1  (ALETRA( 1),A),(ALETRA( 2),B),(ALETRA( 3),C),(ALETRA( 4),D),
     2  (ALETRA( 5),E),(ALETRA( 6),F),(ALETRA( 7),G),(ALETRA( 8),H),
     3  (ALETRA( 9),O),(ALETRA(10),P),(ALETRA(11),Q),(ALETRA(12),R),
     4  (ALETRA(13),S),(ALETRA(14),T),(ALETRA(15),U),(ALETRA(16),V),
     5  (ALETRA(17),W),(ALETRA(18),X),(ALETRA(19),Y),(ALETRA(20),Z)
    1 READ(5,100,END=9) TIRA
      CALL CALCUL
      GO TO 1
    9 PRUB1=(B+O-T)*(C-X)
      PRUB2=DSQRT(DABS(A+D-V))/(F*S)
      PRUB3=(E/H)*(P-G-U)
      PRUB4=(Q-2.*R)/(W+Y-Z)
      WRITE(6,103) PRUB1,PRUB2,PRUB3,PRUB4
      STOP
  100 FORMAT(80A1)
  103 FORMAT(1X,'PRUEBA1=',D16.8,2X,'PRUEBA2=',D16.8,2X,'PRUEBA3=',
     1  D16.8,2X,'PRUEBA4=',D16.8)
      END
C
      SUBROUTINE CALCUL
      IMPLICIT REAL * 8 (A-H,O-Z)
      INTEGER * 2 TIRA,INTER(10),BLANCO,IGUAL,LETRA(20)
      COMMON/CARAC/TIRA(80),ALETRA(20)
      DATA IGUAL/'='/,BLANCO/' '/
      DATA LETRA/'A','B','C','D','E','F','G','H','O','P','Q','R','S',
     1  'T','U','V','W','X','Y','Z'/
      MIN=1
    8 DO 2 I=MIN,80
      IF(TIRA(I).EQ.IGUAL) GO TO 3
    2 CONTINUE
      RETURN
    3 K1=I+1
      KLETRA=I-1
      DO 7 I=K1,80
      IF(TIRA(I).NE.BLANCO) GO TO 10
    7 CONTINUE
   10 K2=I
      DO 4 J=K2,80
      IF(TIRA(J).EQ.BLANCO) GO TO 5
    4 CONTINUE
      GO TO 15
    5 K3=J-1
      CALL ZA03AS(INTER,20)
      WRITE(6,101) (TIRA(L),L=K2,K3)
      CALL ZA03AS(INTER,20)
      READ(5,102) CIFRA
      DO 13 IFOR=1,20
      IF(TIRA(KLETRA).EQ.LETRA(IFOR)) GO TO 12
   13 CONTINUE
   12 ALETRA(IFOR)=CIFRA
      WRITE(6,104) ALETRA(IFOR)
      MIN=K3+1
      GO TO 8
   15 RETURN
  101 FORMAT(16A1)
  102 FORMAT(D16.8)
  104 FORMAT(1X,D16.8)
```

```
      END
/*
//LKED.OTRALIB DD DSN=L10000.HARWELL,DISP=SHR
//LKED.SYSIN DD *,DCB=BLKSIZE=80
          INCLUDE OTRALIB(ZA03AS)
/*
//GO.SYSIN DD *
    A=123456E 03        T=0.3201       G= 541.00574        C=400125.44
R=745.                U=.0                                  V=120.87  X=-4571953E 04
                     E=1.35                               F=8547230.1        P=22.63214
        Q=000251.26        W= +9258147.          S=11.98752    H=875421.23
      D=    -78.254    B=   12.524 Y=       8.125874          Z=    -4.23547 O=-3E-02
/*
//
```

CAP.VII,PROGR.18,CASO 3. Los datos del programa son expresiones mixtas letra-signo igual-número real, que ingresan al vector TIRA declarado INTEGER*2 y de dimensión 80. El juego de datos consta de cualquier cantidad de tarjetas y cada tarjeta de cualquier cantidad de números dispuestos en cualquier orden respecto de las letras que los acompañan y perforados con cualquier formato de los códigos E o F. Puede haber blancos o ceros entre el signo igual y el primer carácter significativo del número (signo +, signo -, punto o dígito), pero no entre los signos aritméticos +/- y el primer dígito. Cada número tiene que terminar seguido de un blanco, inclusive el último dato de una tarjeta. Las expresiones mixtas que constituyen los datos se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      IMPLICIT REAL * 8 (A-H,O-Z)
      INTEGER * 2 TIRA
      DIMENSION TIRA(80),ALETRA(20)
      EQUIVALENCE
     1  (ALETRA( 1),A),(ALETRA( 2),B),(ALETRA( 3),C),(ALETRA( 4),D),
     2  (ALETRA( 5),E),(ALETRA( 6),F),(ALETRA( 7),G),(ALETRA( 8),H),
     3  (ALETRA( 9),O),(ALETRA(10),P),(ALETRA(11),Q),(ALETRA(12),R),
     4  (ALETRA(13),S),(ALETRA(14),T),(ALETRA(15),U),(ALETRA(16),V),
     5  (ALETRA(17),W),(ALETRA(18),X),(ALETRA(19),Y),(ALETRA(20),Z)
    1 READ(5,100,END=9) TIRA
      CALL CALCUL(TIRA,ALETRA)
      GO TO 1
    9 PRUB1=(B+O-T)*(C-X)
      PRUB2=DSQRT(DABS(A+D-V))/(F*S)
      PRUB3=(E/H)*(P-G-U)
      PRUB4=(Q-2.*R)/(W+Y-Z)
      WRITE(6,103) PRUB1,PRUB2,PRUB3,PRUB4
      STOP
  100 FORMAT(80A1)
  103 FORMAT(1X,'PRUEBA1=',D16.8,2X,'PRUEBA2=',D16.8,2X,'PRUEBA3=',
     1  D16.8,2X,'PRUEBA4=',D16.8)
```

```
      END
C
      SUBROUTINE CALCUL(TIRA,ALETRA)
      IMPLICIT REAL * 8 (A-H,O-Z)
      INTEGER * 2 TIRA(80),INTER(10),BLANCO,IGUAL,LETRA(20)
      DIMENSION ALETRA(20)
      DATA IGUAL/'='/,BLANCO/' '/
      DATA LETRA/'A','B','C','D','E','F','G','H','O','P','Q','R','S',
     1  'T','U','V','W','X','Y','Z'/
      MIN=1
    8 DO 2 I=MIN,80
      IF(TIRA(I).EQ.IGUAL) GO TO 3
    2 CONTINUE
      RETURN
    3 K1=I+1
      KLETRA=I-1
      DO 7 I=K1,80
      IF(TIRA(I).NE.BLANCO) GO TO 10
    7 CONTINUE
   10 K2=I
      DO 4 J=K2,80
      IF(TIRA(J).EQ.BLANCO) GO TO 5
    4 CONTINUE
      GO TO 15
    5 K3=J-1
      CALL ZA03AS(INTER,20)
      WRITE(6,101) (TIRA(L),L=K2,K3)
      CALL ZA03AS(INTER,20)
      READ(5,102) CIFRA
      DO 13 IFOR=1,20
      IF(TIRA(KLETRA).EQ.LETRA(IFOR)) GO TO 12
   13 CONTINUE
   12 ALETRA(IFOR)=CIFRA
      WRITE(6,104) ALETRA(IFOR)
      MIN=K3+1
      GO TO 8
   15 RETURN
  101 FORMAT(16A1)
  102 FORMAT(D16.8)
  104 FORMAT(1X,D16.8)
      END
/*
//LKED.OTRALIB DD DSN=L10000.HARWELL,DISP=SHR
//LKED.SYSIN DD *,DCB=BLKSIZE=80
         INCLUDE OTRALIB(ZA03AS)
/*
//GO.SYSIN DD *
    A=123456E 03       T=0.3201      G= 541.00574      C=400125.44
R=745.              U=.0                                V=120.87  X=-4571953E 04
                   E=1.35                            F=8547230.1       P=22.63214
       Q=000251.26       W= +9258147.          S=11.98752   H=875421.23
     D=   -78.254   B=  12.524 Y=      8.125874          Z=   -4.23547 O=-3E-02
/*
//
```

PROGRAMA 19.

OBJETO: Los datos del programa son expresiones mixtas que constan de una letra, el signo igual y un número entero. Estos datos se leen como si fuesen literales y después se extraerá la información numérica convirtiendo en enteros las tiras de caracteres que correspondan, mediante el uso de la rutina ASSEMBLER ZA03AS de la "Harwell Subroutine Library".

CASOS CONSIDERADOS

A) Las expresiones mixtas letra-signo igual-número entero, que constituyen los datos ingresan al programa principal por medio de un formato de lectura.

CASO 1. El programa no tiene rutinas a las cuales se transmitan las expresiones mixtas que constituyen los datos.

CASO 2. Las expresiones mixtas que constituyen los datos se transmiten a una rutina de cálculo por medio de una sentencia COMMON.

CASO 3. Las expresiones mixtas que constituyen los datos se transmiten a una rutina de cálculo como argumentos en la sentencia CALL que la invoca.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

Los datos del programa son los siguientes:

```
M= -645217                    J=-854127          N=     15472
                        L= +31475
          K=    -258                    I=50325417
```

en donde cada renglón representa una tarjeta que se leerá íntegramente como caracteres (incluído el signo igual). La rutina ASSEMBLER ZA03AS de la "Harwell Subroutine Library" se encargará de convertir los caracteres que correspondan en números enteros.

En la marca 1 se leerán tarjetas de datos hasta que la sentencia delimitadora /* transfiera el control del programa a la marca 9 para realizar pruebas numéricas. La tira de caracteres leída ocupará los elementos del vector TIRA declarado INTEGER * 2 y de dimensión 80.

El bucle 8 comprendido entre la marca 8 y la sentencia GO TO 8 leerá todos los datos de cada tarjeta. El DO 2 buscará el signo = del dato; si no lo encontrara (porque todos los datos de esa tarjeta ya fueron procesados) irá a la marca 1 a leer la próxima tarjeta. Si lo encontrara irá a la marca 3: allí establecerá que en K1 empieza el número y que la letra está en KLETRA.

En el DO 7 y a partir de la posición K1 buscará los blancos que pudiera haber entre el igual y el primer carácter significativo del número (el signo+, el signo

- o un dígito). Una vez rechazados todos los blancos irá a la marca 10 en donde establecerá que el primer carácter válido del número está en la posición K2.

El DO 4 buscará el blanco que siempre tendrá que existir al final del número (inclusive detrás del último dato de una tarjeta); cuando lo encuentre irá a la marca 5 donde establecerá que el último dígito del número está en TIRA(K3), que el número consta de una cantidad de dígitos igual a NUMDIG (incluído su propio signo) y que K4 es el número de blancos rechazados.

En este punto el programa llamará a la rutina ASSEMBLER con argumento (INTER,20) en donde INTER es un vector declarado INTEGER*2 y de dimensión 10 porque se supone que el número consta como máximo de 10 caracteres. El número 20 resulta de multiplicar la dimensión de INTER(10) por los 2 octetos de la declaración INTEGER*2; estos 20 octetos representan el espacio que la rutina proveerá en una memoria auxiliar.

Si K4=0 el control del programa pasa a la marca 6 y allí la sentencia

```
  6 WRITE(6,101) (TIRA(LL),LL=K2,K3)
```

escribirá en dicha memoria los caracteres, pero procesados de acuerdo con el formato FORTRAN utilizado y solamente el número propiamente dicho que está comprendido entre TIRA(K2) y TIRA(K3), es decir, un entero sin la letra, sin el signo = y sin blancos, con su propio signo en caso que lo tuviera.

En cambio si K4 no es igual a cero será necesario desplazar los caracteres hacia el octeto de orden inferior, un número de octetos igual a K4 para que el entero que se espera después de la conversión, no resulte multiplicado por la unidad seguida de ceros; en efecto, la sentencia

```
      WRITE(6,101) (ESPACI(IB),IB=1,K4),(TIRA(IC),IC=K2,K3)
```

antes de escribir los caracteres en la memoria auxiliar los desplazará hacia la derecha mediante la inserción de K4 blancos.

Un nuevo llamado a la rutina ASSEMBLER permitirá leer de la memoria auxiliar con la sentencia READ(5,102) ICIFRA, el número entero con el formato FORTRAN utilizado en dicha sentencia.

El DO 13 buscará, por comparación con el DATA LETRA cuál es la letra del alfabeto que está en el elemento TIRA(KLETRA) o sea cuál es la letra que se corresponde con el número ya convertido a entero; cuando la encuentre irá a la marca 12 en donde merced a la sentencia EQUIVALENCE que figura al principio del programa, se establecerá la igualdad entre el número y su letra.

Se corregirá el índice MIN para que busque el dato siguiente en esa misma tarjeta y se volverá a la cabeza del bucle 8.

Cuando todas las tarjetas de datos hayan sido procesadas, el control del programa se transferirá a la marca 9. Allí se realizará una prueba de cálculo que de-

muestre que todas las letras han sido asignadas a sus correspondientes números y que éstos han sido convertidos de manera correcta.

Toda vez que se realizare una comparación de tiras de caracteres en una sentencia IF, las variables, constantes o vectores involucrados en la comparación, deberán ser del mismo tipo; por esta razón han sido explícitamente declarados INTEGER *2.

CAP.VIII,PROGR.19,CASO 1. Los datos del programa son expresiones mixtas letra-signo igual- número entero, que ingresan al vector TIRA declarado INTEGER*2 y de dimensión 80. El juego de datos consta de cualquier cantidad de tarjetas y cada tarjeta de cualquier cantidad de números dispuesto en cualquier orden respecto de las letras que los acompañan y perforados con cualquier formato del código I. Puede haber blancos o ceros entre el signo igual y el primer carácter significativo del número (el signo+, el signo- o 1 dígito), pero no entre los signos aritméticos +/- y el primer dígito. Cada número tiene que terminar seguido de un blanco, inclusive el último dato de una tarjeta. El programa no tiene rutinas a las cuales se transmitan las expresiones mixtas que constituyen los datos.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      INTEGER * 2  TIRA(80),INTER(10),BLANCO,IGUAL,ESPACI(10),LETRA(6)
      DIMENSION IALFA(6)
      EQUIVALENCE
     1  (IALFA(1),I),(IALFA(2),J),(IALFA(3),K),(IALFA(4),L),
     2  (IALFA(5),M),(IALFA(6),N)
      DATA IGUAL/'='/,BLANCO/' '/
      DATA LETRA/'I','J','K','L','M','N'/
      DATA ESPACI/' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' '/
    1 READ(5,100,END=9) TIRA
      MIN=1
    8 DO 2 II=MIN,80
      IF(TIRA(II).EQ.IGUAL) GO TO 3
    2 CONTINUE
      GO TO 1
    3 K1=II+1
      KLETRA=II-1
      DO 7 IA=K1,80
      IF(TIRA(IA).NE.BLANCO) GO TO 10
    7 CONTINUE
   10 K2=IA
      DO 4 JJ=K2,80
      IF(TIRA(JJ).EQ.BLANCO) GO TO 5
    4 CONTINUE
      GO TO 15
    5 K3=JJ-1
      NUMDIG=K3-K2+1
      K4=10-NUMDIG
      CALL ZA03AS(INTER,20)
```

```
      IF(K4.EQ.0) GO TO 6
      WRITE(6,101) (ESPACI(IB),IB=1,K4),(TIRA(IC),IC=K2,K3)
      GO TO 11
    6 WRITE(6,101) (TIRA(LL),LL=K2,K3)
   11 CALL ZA03AS(INTER,20)
      READ(5,102) ICIFRA
      DO 13 IFOR=1,6
      IF(TIRA(KLETRA).EQ.LETRA(IFOR)) GO TO 12
   13 CONTINUE
   12 IALFA(IFOR)= ICIFRA
      WRITE(6,104) IALFA(IFOR)
      MIN=K3+1
      GO TO 8
    9 IPRUB1=K*L
      IPRUB2=J+N-I
      WRITE(6,103) IPRUB1,IPRUB2
   15 STOP
  100 FORMAT(80A1)
  101 FORMAT(10A1)
  102 FORMAT(I10)
  103 FORMAT(1X,'PRUEBA1=',I10,2X,'PRUEBA2=',I10)
  104 FORMAT(1X,I10)
      END
/*
//LKED.OTRALIB DD DSN=L10000.HARWELL,DISP=SHR
//LKED.SYSIN DD *,DCB=BLKSIZE=80
         INCLUDE OTRALIB(ZA03AS)
/*
//GO.SYSIN DD *
   M=  -645217                          J=-854127              N=       15472
                               L=  +31475
             K=      -258                        I=50325417
/*
//
```

CAP.VIII,PROGR.19,CASO 2. Los datos del programa son expresiones mixtas letra-signo igual-número entero, que ingresan al vector TIRA declarado INTEGER*2 y de dimensión 80. El juego de datos consta de cualquier cantidad de tarjetas y cada tarjeta de cualquier cantidad de números dispuestos en cualquier orden respecto de las letras que los acompañan y perforados con cualquier formato del código I. Puede haber blancos o ceros entre el signo igual y el primer carácter significativo del número (el signo+,el signo - o 1 dígito), pero no entre los signos aritméticos +/- y el primer dígito. Cada número tiene que terminar seguido de un blanco, inclusive el último dato de una tarjeta. Las expresiones mixtas que constituyen los datos se transmiten a la SUBROUTINE CALCUL por medio de una sentencia COMMON.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      INTEGER * 2 TIRA
```

```
      COMMON/CARAC/TIRA(80),IALFA(6)
      EQUIVALENCE
     1  (IALFA(1),I),(IALFA(2),J),(IALFA(3),K),(IALFA(4),L),
     2  (IALFA(5),M),(IALFA(6),N)
    1 READ(5,100,END=9) TIRA
      CALL CALCUL
      GO TO 1
    9 IPRUB1=K*L
      IPRUB2=J+N-I
      WRITE(6,103) IPRUB1,IPRUB2
      STOP
  100 FORMAT(80A1)
  103 FORMAT(1X,'PRUEBA1=',I10,2X,'PRUEBA2=',I10)
      END
C
      SUBROUTINE CALCUL
      INTEGER * 2 TIRA,INTER(10),BLANCO,IGUAL,ESPACI(10),LETRA(6)
      COMMON/CARAC/TIRA(80),IALFA(6)
      DATA LETRA/'I','J','K','L','M','N'/
      DATA IGUAL/'='/,BLANCO/' '/
      DATA ESPACI/' ',' ',' ',' ',' ',' ',' ',' ',' ',' '/
      MIN=1
    8 DO 2 II=MIN,80
      IF(TIRA(II).EQ.IGUAL) GO TO 3
    2 CONTINUE
      RETURN
    3 K1=II+1
      KLETRA=II-1
      DO 7 IA=K1,80
      IF(TIRA(IA).NE.BLANCO) GO TO 10
    7 CONTINUE
   10 K2=IA
      DO 4 JJ=K2,80
      IF(TIRA(JJ).EQ.BLANCO) GO TO 5
    4 CONTINUE
      GO TO 15
    5 K3=JJ-1
      NUMDIG=K3-K2+1
      K4=10-NUMDIG
      CALL ZA03AS(INTER,20)
      IF(K4.EQ.0) GO TO 6
      WRITE(6,101) (ESPACI(IB),IB=1,K4),(TIRA(IC),IC=K2,K3)
      GO TO 11
    6 WRITE(6,101) (TIRA(LL),LL=K2,K3)
   11 CALL ZA03AS(INTER,20)
      READ(5,102) ICIFRA
      DO 13 IFOR=1,6
      IF(TIRA(KLETRA).EQ.LETRA(IFOR)) GO TO 12
   13 CONTINUE
   12 IALFA(IFOR)=ICIFRA
      WRITE(6,104) IALFA(IFOR)
      MIN=K3+1
      GO TO 8
   15 RETURN
  100 FORMAT(80A1)
  101 FORMAT(10A1)
  102 FORMAT(I10)
```

```
  104 FORMAT(1X,I10)
      END
/*
//LKED.OTRALIB DD DSN=L10000.HARWELL,DISP=SHR
//LKED.SYSIN DD *,DCB=BLKSIZE=80
         INCLUDE OTRALIB(ZA03AS)
/*
//GO.SYSIN DD *
   M=   -645217                          J=-854127              N=       15472
                               L=  +31475
              K=      -258                        I=50325417
/*
//
```

CAP.VIII,PROGR.19,CASO 3. Los datos del programa son expresiones mixtas letra-signo igual-número entero, que ingresan al vector TIRA declarado INTEGER*2 y de dimensión 80. El juego de datos consta de cualquier cantidad de tarjetas y cada tarjeta de cualquier cantidad de números dispuestos en cualquier orden respecto de las letras que los acompañan y perforados con cualquier formato del código I. Puede haber blancos o ceros entre el signo igual y el primer carácter significativo del número (el signo+,el signo- o 1 dígito), pero no entre los signos aritméticos +/- y el primer dígito. Cada número tiene que terminar seguido de un blanco, inclusive el último dato de una tarjeta. Las expresiones mixtas que constituyen los datos se transmiten a la SUBROUTINE CALCUL como argumentos en la sentencia CALL que la invoca.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      INTEGER * 2 TIRA
      DIMENSION TIRA(80),IALFA(6)
      EQUIVALENCE
     1  (IALFA(1),I),(IALFA(2),J),(IALFA(3),K),(IALFA(4),L),
     2  (IALFA(5),M),(IALFA(6),N)
    1 READ(5,100,END=9) TIRA
      CALL CALCUL(TIRA,IALFA)
      GO TO 1
    9 IPRUB1=K*L
      IPRUB2=J+N-I
      WRITE(6,103) IPRUB1,IPRUB2
      STOP
  100 FORMAT(80A1)
  103 FORMAT(1X,'PRUEBA1=',I10,2X,'PRUEBA2=',I10)
      END
C
      SUBROUTINE CALCUL(TIRA,IALFA)
      INTEGER * 2 TIRA,INTER(10),BLANCO,IGUAL,ESPACI(10),LETRA(6)
      DIMENSION TIRA(80),IALFA(6)
      DATA IGUAL/'='/,BLANCO/' '/
      DATA LETRA/'I','J','K','L','M','N'/
      DATA ESPACI/' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' '/
```

```
      MIN=1
    8 DO 2 II=MIN,80
      IF(TIRA(II).EQ.IGUAL) GO TO 3
    2 CONTINUE
      RETURN
    3 K1=II+1
      KLETRA=II-1
      DO 7 IA=K1,80
      IF(TIRA(IA).NE.BLANCO) GO TO 10
    7 CONTINUE
   10 K2=IA
      DO 4 JJ=K2,80
      IF(TIRA(JJ).EQ.BLANCO) GO TO 5
    4 CONTINUE
      GO TO 15
    5 K3=JJ-1
      NUMDIG=K3-K2+1
      K4=10-NUMDIG
      CALL ZA03AS(INTER,20)
      IF(K4.EQ.0) GO TO 6
      WRITE(6,101) (ESPACI(IB),IB=1,K4),(TIRA(IC),IC=K2,K3)
      GO TO 11
    6 WRITE(6,101) (TIRA(LL),LL=K2,K3)
   11 CALL ZA03AS(INTER,20)
      READ(5,102) ICIFRA
      DO 13 IFOR=1,6
      IF(TIRA(KLETRA).EQ.LETRA(IFOR)) GO TO 12
   13 CONTINUE
   12 IALFA(IFOR)=ICIFRA
      WRITE(6,104) IALFA(IFOR)
      MIN=K3+1
      GO TO 8
   15 RETURN
  100 FORMAT(80A1)
  101 FORMAT(10A1)
  102 FORMAT(I10)
  104 FORMAT(1X,I10)
      END
//LKED.OTRALIB DD DSN=L10000.HARWELL,DISP=SHR
//LKED.SYSIN DD *,DCB=BLKSIZE=80
         INCLUDE OTRALIB(ZA03AS)
/*
//GO.SYSIN DD *
   M=  -645217                          J=-854127                N=       15472
                              L=  +31475
             K=      -258                          I=50325417
/*
//
```

CAPITULO IX

BUSQUEDA CON PROGRAMAS FORTRAN DE TIRAS DE CARACTERES EN LOS DATOS

PROGRAMA 20.

OBJETO: La búsqueda de tiras de caracteres puede tener muy variadas aplicaciones; entre ellas podemos mencionar la siguiente: dadas dos o más versiones antiguas de un programa muy grande reconocer una de ellas por algún detalle estructural.

El programa que busca la tira de caracteres está escrito en lenguaje FORTRAN; los datos del mismo pueden ser a su vez programas escritos en cualquier lenguaje o código, como así también números.

CASOS CONSIDERADOS

A) Los programas-datos se presentan en tarjetas perforadas.

CASO 1. Se busca una tira de 4 caracteres en la columna 2.

CASO 2. Se busca una tira de 7 caracteres en la columna 1.

CASO 3. Se busca una tira de 30 caracteres en la columna 21 (después de la tira hay blancos).

CASO 4. Se busca una tira de 30 caracteres en la columna 21 (después de la tira hay otros caracteres).

CASO 5. Se busca la presencia simultánea de dos tiras de 6 caracteres cada una, ambas en la columna 7.

CASO 6. Se busca la presencia simultánea de una tira de 6 caracteres en la columna 7 y otra tira de 2 caracteres en la columna 53.

CASO 7. Se busca la presencia simultánea de una tira de 3 caracteres en la columna 7 y otra tira de 10 caracteres (seguida de blancos) en la columna 17.

CASO 8. Se busca la presencia simultánea de una tira de 3 caracteres en la columna 7 y otra tira de 10 caracteres (seguida por otros caracteres) en la columna 17.

CASO 9. Se busca la presencia simultánea de una tira de 5 caracteres y otra de 3 caracteres, ambas en la columna 7 (requiere un programa de dos pasos).

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

Se investiga la presencia de la tira de 4 caracteres "1230" en la columna 2. La tira está definida en el DATA de nombre BUSCA.

El bucle 5 comprendido entre la marca 5 y la sentencia GO TO 5 se encargará de leer, con el formato 100 FORMAT(1X,A4), solamente los 4 caracteres de cada tarjeta que aparecen perforados a partir de la columna 2 (el código 1X se saltea una columna), colocándolos en la variable TIRAC.

Para poder realizar la comparación que figura en la sentencia IF, TIRAC y BUSCA

deberán ser del mismo tipo y en efecto ambas son REAL*4 por convención implícita. Si las tiras de caracteres resultaran iguales, el control del programa se transferirá a la marca 2 en donde se escribirá la variable encontrada y desde allí saltará al STOP. Si las tiras de caracteres no fueran iguales el control pasará a la cabeza del bucle 5 para leer la próxima tarjeta.

Si la tira buscada nunca apareciera, cuando la sentencia

```
5 READ(5,100,END=3) TIRAC
```

llegue a la sentencia delimitadora /*, el control será transferido a la marca 3 a escribir una leyenda adecuada y desde allí saltará al STOP.

b) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 2.

Se investiga la presencia de la tira de 7 caracteres "C CANAL" en la columna 1. La tira está definida en el DATA de nombre BUSCA.

El bucle 5 comprendido entre la marca 5 y la sentencia GO TO 5 se encargará de leer, con el formato 100 FORMAT(A7), solamente los 7 caracteres de cada tarjeta que aparecen perforados a partir de la columna 1, colocándolos en la variable TIRAC.

Para poder realizar la comparación que figura en la sentencia IF, TIRAC y BUSCA deberán ser del mismo tipo y en efecto ambas han sido declaradas explícitamente REAL*8. Si las tiras de caracteres resultaran iguales, el control del programa se transferirá a la marca 2 en donde se escribirá la variable encontrada y desde allí saltará al STOP. Si las tiras de caracteres no son iguales el control pasará a la cabeza del bucle 5 para leer la próxima tarjeta.

Si la tira buscada nunca apareciera, cuando la sentencia

```
5 READ(5,100,END=3) TIRAC
```

llegue a la sentencia delimitadora /*, el control será transferido a la marca 3 a escribir una leyenda adecuada y desde allí saltará al STOP.

c) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 3.

Se investiga la presencia de la tira de 30 caracteres

```
"FAC1O(K1212)/(FAC1O(K)*FAC1O1)"
```

a partir de la columna 21. Después del último carácter de la tira, en la tarjeta perforada siguen blancos. La tira está definida en el DATA de nombre BUSCA declarado REAL*8 y de dimensión 4, de la siguiente manera

```
DATA BUSCA/'FAC1O(K1','212)/(FA','C1O(K)*F','AC1O1)  '/
```

En la marca 5 la sentencia

```
5 READ(5,100,END=3) TIRAC
```

leerá en cada tarjeta con el formato 100 FORMAT(20X,4A8), solamente 32 caracteres (4 veces 8 caracteres) de cada tarjeta, a partir de la columna 21 (el código 20X se saltea 20 columnas), colocándolos en el vector TIRAC, declarado REAL*8 y de dimensión 4.

Para poder realizar la comparación que figura en la sentencia IF del DO 1, TIRAC y BUSCA deberán ser del mismo tipo y en efecto ambos son REAL*8 por declaración explícita. Si cualquiera de los elementos del vector TIRAC no es igual al elemento correspondiente del vector BUSCA, las tiras de caracteres no son iguales y el control del programa pasará a la marca 5 a leer la próxima tarjeta. Si el DO 1 se agota, todos los elementos del vector TIRAC son iguales a los correspondientes elementos del vector BUSCA y el control se transferirá a la marca 2 en donde se escribirá la tira de caracteres encontrada y desde allí saltará al STOP.

Si la tira buscada nunca apareciera, cuando la sentencia READ llega a la sentencia delimitadora /*, el control será transferido a la marca 3 a escribir una leyenda adecuada y desde allí saltará al STOP.

d) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 4.

Se investiga la presencia de la tira de 30 caracteres

"FAC10(K1212)/(FAC10(K)*FAC101)"

a partir de la columna 21. Después del último caracter de la tira, en la tarjeta perforada siguen otros caracteres. La tira está definida en dos DATA, el primero de nombre BUSCA1 declarado REAL*8 y de dimensión 3 y el segundo de nombre BUSCA2 declarado REAL*8, de la siguiente manera:

```
DATA BUSCA1/'FAC10(K1','212)/(FA','C10(K)*F'/
DATA BUSCA2/'AC101)'/
```

En la marca 5 la sentencia

```
5 READ(5,100,END=3) TIRAC1,TIRAC2
```

leerá en cada tarjeta, con el formato 100 FORMAT(20X,3A8,A6), 24 caracteres (3 veces 8 caracteres) a partir de la columna 21 (el código 20X se saltea 20 columnas), colocándolos en el vector TIRAC1 declarado REAL*8 y de dimensión 3 y además 6 caracteres que colocará en la variable TIRAC2 declarada REAL*8.

Para poder realizar la comparación que figura en la sentencia IF del DO 1, TIRAC1 y BUSCA1 deberán ser del mismo tipo y la misma exigencia vale para el segundo IF del programa en donde TIRAC2 y BUSCA2 también deberán ser del mismo tipo y en efecto, los 4 son REAL*8 por declaración explícita. Si cualquiera de los elementos del vector TIRAC1 no es igual al elemento correspondiente del vector BUSCA1, las tiras de caracteres no son iguales y el control del programa pasará a la marca 5 a leer la próxima tarjeta. Si el DO 1 se agota, todos los elementos

del vector TIRAC1 son iguales a los correspondientes elementos del vector BUSCA1 y se pasa secuencialmente al segundo IF en donde si TIRAC2 no es igual a BUSCA2, las tiras de caracteres no son enteramente iguales y el control se transferirá a la marca 5 a leer la próxima tarjeta. Si por el contrario son iguales, a su vez las tiras son totalmente iguales y el control del programa pasará a la marca 2 en donde se escribirá la tira de caracteres encontrada y desde allí saltará al STOP.

Si la tira buscada nunca apareciera, cuando la sentencia READ llega a la sentencia delimitadora /*, el control será transferido a la marca 3 a escribir una leyenda adecuada y desde allí saltará al STOP.

e) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 5.

Se investiga la presencia simultánea en el programa-dato de dos tiras de 6 caracteres cada una, ambas en la columna 7. La primera tira está definida en el DATA de nombre BUSCA1 y la segunda en el DATA de nombre BUSCA2 ambas declaradas REAL*8, de la siguiente manera

```
DATA BUSCA1/'VECTOR'/
DATA BUSCA2/'PARAMT'/
```

En la marca 5 la sentencia

```
5 READ(5,100,END=3) TIRAC
```

leerá en cada tarjeta con el formato 100 FORMAT(6X,A6), 6 caracteres a partir de la columna 7 (el código 6X se saltea 6 columnas), colocándolos en la variable TIRAC, REAL*8.

Para poder realizar la comparación que figura en las dos sentencias IF que siguen a la marca 5, las variables TIRAC,BUSCA1 y BUSCA2 deberán ser del mismo tipo y en efecto todas son REAL*8 por declaración explícita.

El bucle 5 comprendido entre la marca 5 y la sentencia GO TO 5 se encargará de señalar la presencia de las tiras investigadas de la siguiente forma: cuando en alguna tarjeta encuentra que TIRAC es igual a BUSCA1, hace IF1=1 y si encuentra en otra tarjeta que TIRAC es igual a BUSCA2, hace IF2=2. La sentencia IF siguiente analiza la presencia de ambas tiras: si las dos han sido encontradas el control del programa pasará a la marca 2 donde avisará esta circunstancia y desde allí saltará al STOP. Si las dos tiras no han sido encontradas todavía, el control volverá a la cabeza del bucle 5 a leer la próxima tarjeta.

Si las tiras buscadas no aparecieran o se encontrara solamente una de ellas, cuando la sentencia READ llega a la sentencia delimitadora /*, el control será transferido a la marca 3. Allí, por medio de sentencias IF y de las igualdades IF1=1 e IF2=2 establecidas anteriormente, el programa avisará si encontró la tira VECTOR solamente o PARAMT solamente o ninguna de las dos y saltará al STOP.

f) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 6.

Se investiga la presencia simultánea en el programa-dato de una tira de 6 caracteres en la columna 7 y otra de 2 caracteres en la columna 53.

La primera tira está definida en el DATA de nombre BUSCA1 declarado REAL*8 y la segunda en el DATA de nombre BUSCA2, REAL*4, de la siguiente manera:

```
DATA BUSCA1/'VECTOR'/
DATA BUSCA2/'SG'/
```

En la marca 5 la sentencia

```
5 READ(5,100,END=3) TIRAC1,TIRAC2
```

leerá en cada tarjeta, con el formato 100 FORMAT(6X,A6,40X,A2), 6 caracteres a partir de la columna 7 (el código 6X se saltea 6 columnas), colocándolos en la variable TIRAC1 declarada REAL*8 y además 2 caracteres a partir de la columna 53 (el código 40X se saltea 40 columnas) que colocará en la variable TIRAC2, REAL*4.

Para poder realizar la comparación que figura en las dos sentencias IF que siguen a la marca 5 , las variables TIRAC1 y BUSCA1 deberán ser del mismo tipo y en efecto ambas son REAL*8 por declaración explícita. La misma exigencia vale para TIRAC2 y BUSCA2 y en efecto ambas son implícitamente REAL*4

El bucle 5 comprendido entre la marca 5 y la sentencia GO TO 5 se encargará de señalar la presencia de las tiras investigadas de la siguiente forma: cuando en alguna tarjeta encuentra que TIRAC1 es igual a BUSCA1,hace IF1=1 y si encuentra en esa misma tarjeta o en otra que TIRAC2 es igual a BUSCA2, hace IF2=2.

La sentencia IF siguiente analiza la presencia de ambas tiras; si las dos han sido encontradas, el control del programa pasará a la marca 2 donde avisará esta circunstancia y desde allí saltará al STOP. Si las dos tiras no han sido encontradas todavía el control volverá a la cabeza del bucle 5 a leer la próxima tarjeta.

Si las tiras buscadas no aparecieran o se encontrara solamente una de ellas, cuando la sentencia READ llega a la sentencia delimitadora /*, el control será transferido a la marca 3. Allí, por medio de sentencias IF y de las igualdades IF1=1 e IF2=2 establecidas anteriormente, el programa avisará si encontró la tira VECTOR solamente o la tira SG solamente o ninguna de las dos y saltará al STOP.

g) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 7.

Se investiga la presencia simultánea en el programa-dato de la tira de 3 caracteres "NOD" a partir de la columna 7 y de la tira de 10 caracteres "ICONT*KAP1" a partir de la columna 17. Después del último carácter de esta segunda tira, en la tarjeta perforada siguen blancos.

La primera tira está definida en el DATA de nombre BUSCA1, REAL*4 y la segunda

en el DATA de nombre BUSCA2 declarado REAL*8 y de dimensión 2, de la siguiente manera:

```
      DATA BUSCA1/'NOD'/
      DATA BUSCA2/'ICONT*KA','P1
```

En la marca 5 la sentencia

```
    5 READ(5,100,END=3) TIRAC1,TIRAC2
```

leerá en cada tarjeta, con el formato 100 FORMAT(6X,A3,7X,2A8), 3 caracteres a partir de la columna 7 (el código 6X se saltea 6 columnas), colocándolos en la variable TIRAC1, REAL*4 y además 16 caracteres (2 veces 8 caracteres) a partir de la columna 17 (el código 7X se saltea 7 columnas) que colocará en el vector TIRAC2 declarado REAL*8 y de dimensión 2.

Para poder realizar la comparación que figura en los dos IF que siguen a la marca 5, las variables TIRAC1 y BUSCA1 deberán ser del mismo tipo y en efecto ambas son implícitamente REAL*4. La misma exigencia vale para los vectores TIRAC2 y BUSCA2 y en efecto ambos son REAL*8 por declaración explícita.

El bucle 5 comprendido entre la marca 5 y la sentencia GO TO 5 se encargará de señalar la presencia de las tiras investigadas de la siguiente forma: cuando en alguna tarjeta encuentra que TIRAC1 es igual a BUSCA1, hace IF1=1 y si encuentra en esa misma tarjeta o en otra que TIRAC2(1) es igual a BUSCA2(1) y TIRAC2(2) es igual a BUSCA2(2), hace IF2=2.

La sentencia IF siguiente analiza la presencia de ambas tiras; si las dos han sido encontradas el control del programa pasará a la marca 2 donde avisará esta circunstancia y desde allí saltará al STOP. Si las dos tiras no han sido encontradas todavía, el control volverá a la cabeza del bucle 5 a leer la próxima tarjeta.

Si las tiras buscadas no aparecieran o se encontrara solamente una de ellas, cuando la sentencia READ llega a la sentencia delimitadora /*, el control será transferido a la marca 3. Allí, por medio de sentencias IF y de las igualdades IF1=1 e IF2=2 establecidas anteriormente, el programa avisará si encontró la tira "NOD" solamente o "ICONT*KAP1" solamente o ninguna de las dos y saltará al STOP.

h) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 8.

Se investiga la presencia simultánea en el programa-dato de la tira de 3 caracteres "NOD" a partir de la columna 7 y de la tira de 10 caracteres "ICONT*KAP*" a partir de la columna 17. Después del último carácter de esta segunda tira, en la tarjeta perforada siguen otros caracteres.

La primera tira está definida en el DATA de nombre BUSCA1, REAL*4 y la segunda tira, parte en el DATA de nombre BUSCA2, declarado REAL*8 y el resto en el DATA

de nombre BUSCA3, REAL*4, en la siguiente forma:

```
DATA BUSCA1/'NOD'/
DATA BUSCA2/'ICONT*KA'/
DATA BUSCA3/'P*'/
```

En la marca 5 de la sentencia

```
5 READ(5,100,END=3) TIRAC1,TIRAC2,TIRAC3
```

leerá en cada tarjeta con el formato 100 FORMAT(6X,A3,7X,A8,A2), 3 caracteres a partir de la columna 7 (el código 6X se saltea 6 columnas), colocándolos en la variable TIRAC1, REAL*4; 8 caracteres a partir de la columna 17 (el código 7X se saltea 7 columnas), colocándolos en la variable TIRAC2, REAL*8 y además 2 caracteres que colocará en la variable TIRAC3, REAL*4.

Para poder realizar la comparación que figura en las dos sentencias IF que siguen a la marca 5, las variables TIRAC1 y BUSCA1, TIRAC3 y BUSCA3 deberán ser del mismo tipo y en efecto todas son implícitamente REAL*4. La misma exigencia vale para las variables TIRAC2 y BUSCA2 y en efecto ambas son REAL*8 por declaración explícita.

El bucle 5 comprendido entre la marca 5 y la sentencia GO TO 5 se encargará de señalar la presencia de las tiras investigadas de la siguiente forma: cuando en alguna tarjeta encuentra que TIRAC1 es igual a BUSCA1, hace IF1=1 y si encuentra en esa misma tarjeta o en otra, que TIRAC2 es igual a BUSCA2 y TIRAC3 es igual a BUSCA3, hace IF2=2.

La sentencia IF siguiente analiza la presencia de ambas tiras; si las dos han sido encontradas el control del programa pasará a la marca 2 donde avisará esta circunstancia y desde allí saltara al STOP. Si las dos tiras no han sido encontradas todavía, el control volverá a la cabeza del bucle 5 a leer la próxima tarjeta.

Si las tiras buscadas no aparecieran o se encontrara solamente una de ellas, cuando la sentencia READ llega a la sentencia delimitadora /*, el control será transferido a la marca 3. Allí, por medio de sentencias IF y de las igualdades IF1=1 e IF2=2 establecidas anteriormente, el programa avisará si encontró la tira "NOD" solamente o "ICONT*KAP*" solamente o ninguna de las dos y saltará al STOP.

i) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 9.

Se investiga la presencia simultánea en el programa-dato de una tira de 5 caracteres y otra de 3 caracteres, ambas en la columna 7. El problema se puede resolver con un programa de dos pasos: en el PASO 1 se busca la tira de 5 caracteres y en el PASO 2 la de 3. Una Referencia hacia Atrás en la etapa GO permitirá utilizar en el segundo paso los datos ya leídos en el primero.

PASO 1. Se investiga la presencia de la tira de 5 caracteres "CANAL" en la columna 7. La tira está definida en el DATA de nombre BUSCA.

El bucle 5 comprendido entre la marca 5 y la sentencia GO TO 5 se encargará de leer con el formato 100 FORMAT(6X,A5), solamente los 5 caracteres de cada tarjeta que aparecen perforados a partir de la columna 7 (el código 6X se saltea 6 columnas), colocándolos en la variable TIRAC.

Para poder realizar la comparación que figura en la sentencia IF, las variables TIRAC y BUSCA deberán ser del mismo tipo y en efecto ambas son REAL*8 por declaración explícita. Si las tiras de caracteres resultaran iguales, el control del programa se transferirá a la marca 2 en donde se escribirá la variable encontrada y desde allí saltará al STOP. Si las tiras no fueran iguales el control pasará a la cabeza del bucle 5 para leer la próxima tarjeta.

Si la tira buscada nunca apareciera, cuando la sentencia

```
5 READ(5,100,END=3) TIRAC
```

llega a la sentencia delimitadora /*, el control será transferido a la marca 3 a escribir una leyenda adecuada y desde allí saltará al STOP.

PASO 2. Se investiga la presencia de la tira de 3 caracteres "MAS" en la columna 7. La tira está definida en el DATA de nombre BUSCA.

El bucle 5 comprendido entre la marca 5 y la sentencia GO TO 5 se encargará de leer con el formato 100 FORMAT(6X,A3), solamente los 3 caracteres de cada tarjeta que aparecen perforados a partir de la columna 7 (el código 6X se saltea 6 columnas), colocándolos en la variable TIRAC.

El resto del programa es igual al PASO 1.

CAP.IX,PROGR.20,CASO 1. Se investiga la presencia de la tira de 4 caracteres 1230 en la columna 2. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA BUSCA/'1230'/
    5 READ(5,100,END=3) TIRAC
      IF(TIRAC.EQ.BUSCA) GO TO 2
      GO TO 5
    3 WRITE(6,102)
      GO TO 4
    2 WRITE(6,101) TIRAC
    4 STOP
  100 FORMAT(1X,A4)
  101 FORMAT(1X,' ENCONTRO LA TIRA DE CARACTERES: ',A4,'.EL PROGRAMA ES
     1LA VERSION DEL 15/12/81')
  102 FORMAT(1X,' LA TIRA DE CARACTERES NO EXISTE. EL PROGRAMA ES LA VER
     1SION DEL 8/3/82')
      END
```

```
/*
//GO.SYSIN DD *
 Aquí tarjetas perforadas del programa en el cual se busca la tira
 de caracteres
/*
//
```

CAP.IX,PROGR.20,CASO 2. Se investiga la presencia de la tira de 7 caracteres "C CANAL" en la columna 1. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 TIRAC,BUSCA
      DATA BUSCA/'C CANAL'/
    5 READ(5,100,END=3) TIRAC
      IF(TIRAC.EQ.BUSCA) GO TO 2
      GO TO 5
    3 WRITE(6,102)
      GO TO 4
    2 WRITE(6,101) TIRAC
    4 STOP
  100 FORMAT(A7)
  101 FORMAT(1X,' ENCONTRO LA TIRA DE CARACTERES: ',A7,'.EL PROGRAMA ES
     1LA VERSION DEL 15/12/81')
  102 FORMAT(1X,' LA TIRA DE CARACTERES NO EXISTE. EL PROGRAMA ES LA VER
      SION DEL 8/3/82')
      END
/*
//GO.SYSIN DD *
 Aquí tarjetas perforadas del programa en el cual se busca la tira
 de caracteres
/*
//
```

CAP.IX,PROGR.20,CASO 3. Se investiga la presencia de la tira de 30 caracteres "FAC10(K1212)/(FAC10(K)*FAC101)" en la columna 21. Después de la tira propuesta siguen blancos. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 TIRAC,BUSCA
      DIMENSION TIRAC(4),BUSCA(4)
      DATA BUSCA/'FAC10(K1','212)/(FA','C10(K)*F','AC101)  '/
    5 READ(5,100,END=3) TIRAC
      DO 1 I=1,4
      IF(TIRAC(I).NE.BUSCA(I)) GO TO 5
    1 CONTINUE
      GO TO 2
    3 WRITE(6,102)
      GO TO 4
    2 WRITE(6,101) TIRAC
```

```
     4 STOP
   100 FORMAT(20X,4A8)
   101 FORMAT(1X,' ENCONTRO LA TIRA DE CARACTERES: ',4A8,'. EL PROGRAMA E
      1S LA VERSION DEL 15/12/81')
   102 FORMAT(1X,' LA TIRA DE CARACTERES NO EXISTE. El PROGRAMA ES LA VER
      1SION DEL 8/3/82')
       END
/*
//GO.SYSIN DD *
  Aquí tarjetas perforadas del programa en el cual se busca la tira
  de caracteres
/*
//
```

CAP.IX,PROGR.20,CASO 4. Se investiga la presencia de la tira de 30 caracteres "FAC10(K1212)/(FAC10(K)*FAC101)" en la columna 21. Después de la tira propuesta siguen otros caracteres. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
       REAL * 8 TIRAC1,TIRAC2,BUSCA1,BUSCA2
       DIMENSION TIRAC1(3),BUSCA1(3)
       DATA BUSCA1/'FAC10(K1','212)/(FA','C10(K)*F'/
       DATA BUSCA2/'AC101)'/
     5 READ(5,100,END=3) TIRAC1,TIRAC2
       DO 1 I=1,3
       IF(TIRAC1(I).NE.BUSCA1(I)) GO TO 5
     1 CONTINUE
       IF(TIRAC2.NE.BUSCA2) GO TO 5
       GO TO 2
     3 WRITE(6,102)
       GO TO 4
     2 WRITE(6,101) TIRAC1,TIRAC2
     4 STOP
   100 FORMAT(20X,3A8,A6)
   101 FORMAT(1X,' ENCONTRO LA TIRA DE CARACTERES: ',3A8,A6,'.EL PROGRAMA
      1 ES LA VERSION DEL 15/12/81')
   102 FORMAT(1X,' LA TIRA DE CARACTERES NO EXISTE. EL PROGRAMA ES LA VER
      1SION DEL 8/3/82')
       END
/*
//GO.SYSIN DD *
  Aquí tarjetas perforadas del programa en el cual se busca la tira
  de caracteres
/*
//
```

CAP.IX,PROGR.20,CASO 5. Se investiga la presencia simultánea en el programa-dato de las tiras de 6 caracteres "VECTOR" y "PARAMT", ambas en la columna 7. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
```

```
//FORT.SYSIN DD *
      REAL * 8 TIRAC,BUSCA1,BUSCA2
      DATA BUSCA1/'VECTOR'/
      DATA BUSCA2/'PARAMT'/
      IF1=0
      IF2=0
    5 READ(5,100,END=3) TIRAC
      IF(TIRAC.EQ.BUSCA1) IF1=1
      IF(TIRAC.EQ.BUSCA2) IF2=2
      IF(IF1.EQ.1.AND.IF2.EQ.2) GO TO 2
      GO TO 5
    3 IF(IF1.NE.1.AND.IF2.NE.2) GO TO 7
      IF(IF1.EQ.1.AND.IF2.NE.2) GO TO 8
      IF(IF1.NE.1.AND.IF2.EQ.2) GO TO 9
    7 WRITE(6,101)
      GO TO 4
    8 WRITE(6,102)
      GO TO 4
    9 WRITE(6,103)
      GO TO 4
    2 WRITE(6,104)
    4 STOP
  100 FORMAT(6X,A6)
  101 FORMAT(1X,' NO ENCONTRO NINGUNA DE LAS VARIABLES')
  102 FORMAT(1X,' ENCONTRO LA VARIABLE VECTOR PERO NO LA VARIABLE PARAMT
     1')
  103 FORMAT(1X,' ENCONTRO LA VARIABLE PARAMT PERO NO LA VARIABLE VECTOR
     1')
  104 FORMAT(1X,' ENCONTRO LAS DOS VARIABLES. EL PROGRAMA ES LA VERSION
     1DEL 3/12/81')
      END
/*
//GO.SYSIN DD *
 Aquí tarjetas perforadas del programa en el cual se buscan las tiras
 de caracteres
/*
//
```

NOTA: En el CASO 9 se muestra cómo se puede resolver este problema con un programa de dos pasos si las tiras de caracteres estando en la misma columna no tuvieran el mismo número de caracteres.

CAP.IX,PROGR.20,CASO 6. Se investiga la presencia simultánea en el programa-dato de la tira de 6 caracteres "VECTOR" en la columna 7 y de la tira de 2 caracteres "SG" en la columna 53. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 TIRAC1,BUSCA1
      DATA BUSCA1/'VECTOR'/
      DATA BUSCA2/'SG'/
      IF1=0
      IF2=0
```

```
    5 READ(5,100,END=3) TIRAC1,TIRAC2
      IF(TIRAC1.EQ.BUSCA1) IF1=1
      IF(TIRAC2.EQ.BUSCA2) IF2=2
      IF(IF1.EQ.1.AND.IF2.EQ.2) GO TO 2
      GO TO 5
    3 IF(IF1.NE.1.AND.IF2.NE.2) GO TO 7
      IF(IF1.EQ.1.AND.IF2.NE.2) GO TO 8
      IF(IF1.NE.1.AND.IF2.EQ.2) GO TO 9
    7 WRITE(6,101)
      GO TO 4
    8 WRITE(6,102)
      GO TO 4
    9 WRITE(6,103)
      GO TO 4
    2 WRITE(6,104)
    4 STOP
  100 FORMAT(6X,A6,40X,A2)
  101 FORMAT(1X,' NO ENCONTRO NINGUNA DE LAS VARIABLES')
  102 FORMAT(1X,' ENCONTRO LA VARIABLE VECTOR PERO NO LA VARIABLE SG')
  103 FORMAT(1X,' ENCONTRO LA VARIABLE SG PERO NO LA VARIABLE VECTOR')
  104 FORMAT(1X,' ENCONTRO LAS DOS VARIABLES. EL PROGRAMA ES LA VERSION
     1DEL 3/12/81')
      END
/*
//GO.SYSIN DD *
 Aquí las tarjetas perforadas del programa en el cual se buscan las tiras
 de caracteres
/*
//
```

CAP.IX,PROGR.20,CASO 7. Se investiga la presencia simultánea en el programa-dato de la tira de 3 caracteres "NOD" a partir de la columna 7 y de la tira de 10 caracteres "ICONT*KAP1" a partir de la columna 17. Después del último carácter de esta segunda tira, en la tarjeta perforada siguen blancos. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 TIRAC2,BUSCA2
      DIMENSION TIRAC2(2),BUSCA2(2)
      DATA BUSCA1/'NOD'/
      DATA BUSCA2/'ICONT*KA','P1      '/
      IF1=0
      IF2=0
    5 READ(5,100,END=3) TIRAC1,TIRAC2
      IF(TIRAC1.EQ.BUSCA1) IF1=1
      IF(TIRAC2(1).EQ.BUSCA2(1).AND.TIRAC2(2).EQ.BUSCA2(2)) IF2=2
      IF(IF1.EQ.1.AND.IF2.EQ.2) GO TO 2
      GO TO 5
    3 IF(IF1.NE.1.AND.IF2.NE.2) GO TO 7
      IF(IF1.EQ.1.AND.IF2.NE.2) GO TO 8
      IF(IF1.NE.1.AND.IF2.EQ.2) GO TO 9
```

```
    7 WRITE(6,101)
      GO TO 4
    8 WRITE(6,102)
      GO TO 4
    9 WRITE(6,103)
      GO TO 4
    2 WRITE(6,104)
    4 STOP
  100 FORMAT(6X,A3,7X,2A8)
  101 FORMAT(1X,' NO ENCONTRO NINGUNA DE LAS TIRAS DE CARACTERES')
  102 FORMAT(1X,' ENCONTRO LA TIRA DE TRES CARACTERES PERO NO LA TIRA DE
     1 DIEZ CARACTERES')
  103 FORMAT(1X,' ENCONTRO LA TIRA DE DIEZ CARACTERES PERO NO LA TIRA DE
     1 TRES CARACTERES')
  104 FORMAT(1X,' ENCONTRO LAS DOS TIRAS DE CARACTERES BUSCADAS. EL PROG
     1RAMA ES LA VERSION DEL 3/12/81')
      END
/*
//GO.SYSIN DD *
 Aquí tarjetas perforadas del programa en el cual se buscan las tiras
 de caracteres
/*
//
```

CAP.IX,PROGR.20,CASO 8. Se investiga la presencia simultánea en el programa-dato de la tira de 3 caracteres "NOD" a partir de la columna 7 y de la tira de 10 caracteres "ICONT*KAP*" a partir de la columna 17. Después del último carácter de esta segunda tira, en la tarjeta siguen otros caracteres. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 TIRAC2,BUSCA2
      DATA BUSCA1/'NOD'/
      DATA BUSCA2/'ICONT*KA'/
      DATA BUSCA3/'P*'/
      IF1=0
      IF2=0
    5 READ(5,100,END=3) TIRAC1,TIRAC2,TIRAC3
      IF(TIRAC1.EQ.BUSCA1) IF1=1
      IF(TIRAC2.EQ.BUSCA2.AND.TIRAC3.EQ.BUSCA3) IF2=2
      IF(IF1.EQ.1.AND.IF2.EQ.2) GO TO 2
      GO TO 5
    3 IF(IF1.NE.1.AND.IF2.NE.2) GO TO 7
      IF(IF1.EQ.1.AND.IF2.NE.2) GO TO 8
      IF(IF1.NE.1.AND.IF2.EQ.2) GO TO 9
    7 WRITE(6,101)
      GO TO 4
    8 WRITE(6,102)
      GO TO 4
    9 WRITE(6,103)
      GO TO 4
    2 WRITE(6,104)
```

```
    4 STOP
  100 FORMAT(6X,A3,7X,A8,A2)
  101 FORMAT(1X,' NO ENCONTRO NINGUNA DE LAS TIRAS DE CARACTERES')
  102 FORMAT(1X,' ENCONTRO LA TIRA DE TRES CARACTERES PERO NO LA TIRA DE
     1 DIEZ CARACTERES')
  103 FORMAT(1X,' ENCONTRO LA TIRA DE DIEZ CARACTERES PERO NO LA TIRA DE
     1 TRES CARACTERES')
  104 FORMAT(1X,' ENCONTRO LAS DOS TIRAS DE CARACTERES BUSCADAS. EL PROG
     1RAMA ES LA VERSION DEL 3/12/81')
      END
/*
//GO.SYSIN DD *
 Aquí tarjetas perforadas del programa en el cual se buscan las tiras
 de caracteres
/*
//
```

CAP.IX,PROGR.20,CASO 9. Se investiga la presencia simultánea en el programa-dato de la tira de 5 caracteres "CANAL" y de la tira de 3 caracteres "MAS", ambas en la columna 7. Requiere un programa de dos pasos. El programa-dato se presenta en tarjetas que se leen una sola vez.

```
// .... JOB ........
//PASO1  EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      REAL * 8 TIRAC,BUSCA
      DATA BUSCA/'CANAL'/
    5 READ(5,100,END=3) TIRAC
      IF(TIRAC.EQ.BUSCA) GO TO 2
      GO TO 5
    3 WRITE(6,102)
      GO TO 4
    2 WRITE(6,101) TIRAC
    4 STOP
  100 FORMAT(6X,A5)
  101 FORMAT(1X,' ENCONTRO LA TIRA DE CARACTERES: ',A5)
  102 FORMAT(1X,' LA TIRA DE CARACTERES NO EXISTE')
      END
/*
//GO.SYSIN DD *
 Aquí tarjetas perforadas del programa en el cual se buscan las tiras
 de caracteres
/*
//PASO2 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA BUSCA/'MAS'/
    5 READ(5,100,END=3) TIRAC
      IF(TIRAC.EQ.BUSCA) GO TO 2
      GO TO 5
    3 WRITE(6,102)
      GO TO 4
    2 WRITE(6,101) TIRAC
    4 STOP
  100 FORMAT(6X,A3)
```

```
  101 FORMAT(1X,' ENCONTRO LA TIRA DE CARACTERES: ',A3)
  102 FORMAT(1X,' LA TIRA DE CARACTERES NO EXISTE')
      END
/*
//GO.SYSIN DD DSNAME=*.PASO1.GO.FT05F001,DISP=(OLD,PASS)
/*
//
```

PROGRAMA 21.

OBJETO: Mostrar el procedimiento a seguir cuando el programa-dato no se presenta en tarjetas perforadas sino que reside en disco, en una biblioteca particionada y catalogada de programas en lenguaje simbólico. Se toma como ejemplo el CASO 1 del PROGRAMA 20, ya descripto al principio de este capítulo y se transcribe el modelo completo con todas las sentencias de lenguaje de control necesarias.

CASOS CONSIDERADOS

A) Los programas-datos residen en una biblioteca.

CASO 1. Se busca una tira de 4 caracteres en la columna 2.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

El programa consta de dos pasos. En el PASO 1 el utilitario IEBGENER extrae de la biblioteca L06203.SIMBFIS el miembro de nombre ADRALFA1 en el cual se buscará la tira de caracteres "1230" en la columna 2 de una imagen de tarjeta, cargándolo en un disco auxiliar. En el PASO 2 el procedimiento FTG1CLG ejecuta el programa FORTRAN encargado de buscar la tira de caracteres, leyendo el programa-dato en la etapa GO directamente del disco auxiliar en el cual fuera grabado en el paso anterior.

CAP.IX,PROGR.21,CASO 1. Modelo para ejecutar el CASO 1 del PROGRAMA 20 cuando el programa-dato reside en una biblioteca particionada y catalogada de nombre L06203.SIMBFIS.

```
// .... JOB ........
//PASO1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN DD DUMMY
//SYSUT1 DD DSN=L06203.SIMBFIS(ADRALFA1),DISP=SHR
//SYSUT2 DD DSN=ADRALFA1,UNIT=WORKDISK,DISP=(NEW,PASS),
//  SPACE=(CYL,(1,1)),DCB=(BLKSIZE=3520,LRECL=80,RECFM=FB)
/*
//PASO2 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA BUSCA/'1230'/
    5 READ(9,100,END=3) TIRAC
      IF(TIRAC.EQ.BUSCA) GO TO 2
      GO TO 5
    3 WRITE(6,102)
      GO TO 4
    2 WRITE(6,101) TIRAC
    4 STOP
  100 FORMAT(1X,A4)
```

```
  101 FORMAT(1X,' ENCONTRO LA TIRA DE CARACTERES: ',A4,'.EL PROGRAMA ES
     1LA VERSION DEL 15/12/81')
  102 FORMAT(1X,' LA TIRA DE CARACTERES NO EXISTE. EL PROGRAMA ES LA VER
     1SION DEL 8/3/82')
      END
/*
//GO.FT09F001 DD DSN=ADRALFA1,UNIT=WORKDISK,DISP=(OLD,DELETE)
/*
//
```

PROGRAMA 22.

OBJETO: Mostrar el procedimiento a seguir cuando el programa-dato no se presenta en tarjetas perforadas sino que reside en una cinta magnética. Se toma como ejemplo el CASO 1 del PROGRAMA 20, ya descripto al principio de este capítulo y se transcribe el modelo completo con todas las sentencias de lenguaje de control necesarias.

CASOS CONSIDERADOS

A) Los programas-datos están grabados en cintas magnéticas.

CASO 1. Se busca una tira de 4 caracteres en la columna 2.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

En el PASO 1 el procedimiento FTG1CLG ejecuta el programa FORTRAN encargado de buscar la tira de caracteres "1230" en la columna 2 de una imagen de tarjeta, en el programa de nombre ADRALFA1 que está grabado con el DCB=(BLKSIZE=3200,LRECL=80, RECFM=FB), en el rótulo 13 de una cinta magnética NL (NO LABEL) que lleva el número de carrete T80371. En la etapa GO el programa-dato se lee directamente de la cinta.

CAP.IX,PROGR.22,CASO 1. Modelo para ejecutar el CASO 1 del PROGRAMA 20 cuando el programa-dato está grabado en una cinta magnética NO LABEL.

```
// .... JOB ........
//PASO1 EXEC PROC=FTG1CLG
//FORT.SYSIN DD *
      DATA BUSCA/'1230'/
    5 READ(9,100,END=3) TIRAC
      IF(TIRAC.EQ.BUSCA) GO TO 2
      GO TO 5
    3 WRITE(6,102)
      GO TO 4
    2 WRITE(6,101) TIRAC
    4 STOP
  100 FORMAT(1X,A4)
  101 FORMAT(1X,' ENCONTRO LA TIRA DE CARACTERES: ',A4,'.EL PROGRAMA ES
     1LA VERSION DEL 15/12/81')
  102 FORMAT(1X,' LA TIRA DE CARACTERES NO EXISTE. EL PROGRAMA ES LA VER
     1SION DEL 8/3/82')
      END
/*
//GO.FT09F001 DD UNIT=TAPE,VOL=SER=T80371,DISP=(OLD,KEEP),
//  LABEL=(13,NL,,IN),DCB=(BLKSIZE=3200,LRECL=80,RECFM=FB)
/*
//
```

CAPITULO X

BUSQUEDA CON EL UTILITARIO OSLISTA DE TIRAS DE CARACTERES EN LOS DATOS

PROGRAMA 23.

OBJETO: La búsqueda de tiras de caracteres puede tener muy variadas aplicaciones; entre ellas podemos mencionar la siguiente: dadas dos o más versiones antiguas de un programa muy grande, reconocer una de ellas por algún detalle estructural. El programa que busca la tira de caracteres es el utilitario OSLISTA; los datos del mismo pueden ser a su vez programas escritos en cualquier lenguaje o código, como así también números.

CASOS CONSIDERADOS

A) Los programas-datos se presentan en tarjetas perforadas.

CASO 1. Se busca una tira de 4 caracteres en la columna 2.
CASO 2. Se busca una tira de 30 caracteres en la columna 21.
CASO 3. Se busca la presencia simultánea en el programa-dato de una tira de 6 caracteres en la columna 7 y otra de 2 caracteres en la columna 53.
CASO 4. Se busca la presencia simultánea en el programa-dato de una tira de 5 caracteres y otra de 3 caracteres, ambas en la columna 7.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

El programa OSLISTA es un utilitario escrito en 1971 por el Analista de Sistemas Sr. Jorge H. Vattuone, del Centro Unico de Procesamiento Electrónico de Datos (CUPED). Entre las tarjetas de control usaremos las siguientes:

//SYSUT1 DD *	define los datos de entrada o sea el programa en el cual se buscará la tira de caracteres.
//SYSUT2 DD *	define las características de la salida impresa.
RCIN N,,'XXXXX'	N es un número comprendido entre 1 y 80 que indica en cuál columna de la tarjeta empieza la tira de caracteres que se busca (en nuestro caso será N=2). Los caracteres buscados son los que figuran entre apóstrofos (en nuestro caso la tira 1230).
PRINT FL=C	indica que todo el registro será impreso con el formato C (caracteres).

El programa tiene otras muchas facilidades que no necesitamos utilizar.

CAP.X,PROGR.23,CASO 1. Se investiga la presencia de la tira de 4 caracteres

"1230" en la columna 2. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PGM=OSLISTA
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD SYSOUT=A,DCB=(BLKSIZE=3990,LRECL=133,RECFM=FB)
//SYSUT1 DD *
 Aquí tarjetas perforadas del programa en el cual se busca la tira
 de caracteres
//SYSIN DD *
RCIN 2,,'1230'
PRINT FL=C
/*
//
```

CAP.X,PROGR.23,CASO 2. Se investiga la presencia de la tira de 30 caracteres "FAC10(K1212)/(FAC10(K)*FAC101)" en la columna 21. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PGM=OSLISTA
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD SYSOUT=A,DCB=(BLKSIZE=3990,LRECL=133,RECFM=FB)
//SYSUT1 DD *
 Aquí tarjetas perforadas del programa en el cual se busca la tira
 de caracteres
//SYSIN DD *
RCIN 21,,'FAC10(K1212)/(FAC10(K)*FAC101)'
PRINT FL=C
/*
//
```

CAP.X,PROGR.23,CASO 3. Se investiga la presencia simultánea en el programa-dato de la tira de 2 caracteres "SG" en la columna 53 y la tira de 6 caracteres "VECTOR" en la columna 7. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PGM=OSLISTA
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD SYSOUT=A,DCB=(BLKSIZE=3990,LRECL=133,RECFM=FB)
//SYSUT1 DD *
 Aquí tarjetas perforadas del programa en el cual se buscan las tiras
 de caracteres.
//SYSIN DD *
RCIN 53,,'SG'
RCIN 7,,'VECTOR'
PRINT FL=C
/*
//
```

CAP.X,PROGR.23,CASO 4. Se investiga la presencia simultánea en el programa-dato

de la tira de 5 caracteres "CANAL" y la tira de 3 caracteres "MAS", ambas en la columna 7. El programa-dato se presenta en tarjetas.

```
// .... JOB ........
//PASO1 EXEC PGM=OSLISTA
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD SYSOUT=A,DCB=(BLKSIZE=3990,LRECL=133,RECFM=FB)
//SYSUT1 DD *
 Aquí tarjetas perforadas del programa en el cual se buscan las tiras
 de caracteres
//SYSIN DD *
RCIN 7,,'CANAL'
RCIN 7,,'MAS'
PRINT FL=C
/*
//
```

PROGRAMA 24.

OBJETO: Mostrar el procedimiento a seguir cuando el programa-dato no se presenta en tarjetas perforadas sino que reside en disco, en una biblioteca particionada y catalogada de programas en lenguaje simbólico. Se toma como ejemplo el CASO 1 del PROGRAMA 23, ya descripto al principio de este capítulo y se transcribe el modelo completo con todas las sentencias de lenguaje de control necesarias.

CASOS CONSIDERADOS

A) Los programas-datos residen en una biblioteca.

CASO 1. Se busca una tira de 4 caracteres en la columna 2.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

El utilitario OSLISTA, por medio de la sentencia //SYSUT1 extraerá de la biblioteca L06203.SIMBFIS el miembro de nombre ADRALFA1, en el cual se buscará la tira de 4 caracteres "1230" en la columna 2.

CAP.X,PROGR.24,CASO 1. Modelo para ejecutar el CASO 1 del PROGRAMA 23 cuando el programa-dato reside en una biblioteca particionada y catalogada de nombre L06203.SIMBFIS.

```
// .... JOB ........
//PASO1 EXEC PGM=OSLISTA
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD SYSOUT=A,DCB=(BLKSIZE=3990,LRECL=133,RECFM=FB)
//SYSUT1 DD DSN=L06203.SIMBFIS(ADRALFA1),DISP=SHR
//SYSIN DD *
RCIN 2,,'1230'
PRINT FL=C
/*
//
```

PROGRAMA 25.

OBJETO: Mostrar el procedimiento a seguir cuando el programa-dato no se presenta en tarjetas perforadas sino que reside en una cinta magnética. Se toma como ejemplo el CASO 1 del PROGRAMA 23, ya descripto al principio de este capítulo y se transcribe el modelo completo con todas las sentencias de lenguaje de control necesarias.

CASOS CONSIDERADOS

A) Los programas-datos están grabados en cintas magnéticas.

CASO 1. Se busca una tira de 4 caracteres en la columna 2.

a) DESCRIPCION DEL PROGRAMA UTILIZADO COMO EJEMPLO EN EL CASO 1.

El utilitario OSLISTA busca la tira de 4 caracteres "1230" en la columna 2 de una imagen de tarjeta del programa ADRALFA1 que está grabado, con el DCB=(BLKSIZE=3200,LRECL=80,RECFM=FB), en el rótulo 13 de una cinta magnética NL (NO LABEL) que lleva el número de carrete T80371.

CAP.X,PROGR.25,CASO 1. Modelo para ejecutar el CASO 1 del PROGRAMA 23 cuando el programa-dato está grabado en una cinta magnética NO LABEL.

```
// .... JOB ........
//PASO1 EXEC PGM=OSLISTA
//SYSPRINT DD SYSOUT=A
//SYSUT2 DD SYSOUT=A,DCB=(BLKSIZE=3990,LRECL=133,RECFM=FB)
//SYSUT1 DD UNIT=TAPE,VOL=SER=T80371,DISP=(OLD,KEEP),
//  LABEL=(13,NL,,IN),DCB=(BLKSIZE=3200,LRECL=80,RECFM=FB)
//SYSIN DD *
RCIN 2,,'1230'
PRINT FL=C
/*
//
```

IMPRESO EN CNEA - ABRIL 1982