NOTAS TECNICAS SOBRE EL SISTEMA

1.- Modulos externos en PASCAL

Para compilar un programa y crear un código objeto, es necesario darle un nombre, que sirve para identificarle posteriormente. Esto puede hacerse mediante una tarjeta en JCL de la forma siguiente:

```
//PASC.SYSGO    DD    DSN=&&MODULO
```

En el programa donde se hace referencia al procedimiento externo debe aparecer, en las tarjetas de link-edición la definición de dicho modulo.

A continuación puede verse un ejemplo:

```
//  EXEC     PASC
//PASC.SYSGO   DD   DSN=&&MODULO
//PASC.SYSIN   DD   *
          (* & E + * )
          PROGRAM  EXTERNO  (OUTPUT)  ;                    Compilación del pro-
               PROCEDURE  ESCRIBIR  (N:INTEGER) ;          grama EXTERNO y crea-
               BEGIN                                       cion del programa ob-
               WRITELN  ('ESCRIBIR N', N)                  jeto con el nombre
                    END      ;                             MODULO
//SEGPAS  EXEC   PASCLG
//PASC.SYSIN   DD   *
          PROGRAM   PRUEBA   (INPUT) ;
               PROCEDURE  ESCRIBIR  (J:INTEGER) ; PASCAL ; (*referencia al
               BEGIN                                         modulo externo*)
                    READ  (J) ;
                    ESCRIBIR
               END.
//LKED.SYSLIN  DD  DISP=SHR,DSN=SYS1.PROCLIB(PASCLCC)
// DD   DISP=(OLD,PASS),DSN=&&OBJ                          linkedición del pro-
// DD   DISP=(OLD,PASS),DSN=&&MODULO                       grama con el modulo
                                                           externo
```

2. Utilización del PL/1 Optimizer

El PL/1 Optimizer está implementado en nuestro sistema. El objetivo de este compilador es producir un código objeto lo más eficiente posible. Para ello existe una opción (OPTIMIZE) que puede especificarse como parámetro de la tarjeta EXEC. En este caso el tiempo de compilación será más alto, mejorando el tiempo de ejecución. Los procedimientos catalogados para la llamada a este compilador son los siguientes:

```
//  EXEC   PLIXC      para compilar
//  EXEC   PLIXCL     para compilar y linkeditar
//  EXEC   PLIXCLG    para compilar, linkeditar y ejecutar
//  EXEC   PLIXCG     para compilar,cargar y ejecutar
```

El PL/1 Optimizer presenta ciertas extensiones frente al PL/1 F, entre ellas podemos citar:

Atributos

ENTRY - pueden declararse variables con este atributo para ello hay que anadir la palabra VARIABLE por ejemplo:

```
DECLARE CONTROL ENTRY VARIABLE
```

es una instrucción en la que declaramos CONTROL como una variable que puede tomar como valores nombres ENTRY.

```
si tenemos      DECLARE CONTROL ENTRY VARIABLE,
                    (P1, P2) ENTRY  ;
                          .
                          .
                          .
y la instrucción  CONTROL=P1 ;   es posible hacer la llamada
                          .
                          .
                          .
                  CALL  CONTROL ;
                          .
                          .
                          .
```

Instrucciones

DEFAULT - da al programador control sobre los atributos por defecto que se asignan a los identificadores.
Por ejemplo: DEFAULT RANGE (P) POINTER ; hace que cualquier identificador que se encuentre en el ambito de esta instrucción, y que empiece por la letra P, se le asigne el atributo POINTER.

FETCH (nombre)
Copia el procedimiento (nombre) de un almacenamiento externo a la memoria principal para ser utilizada posteriormente en una instrucción CALL.

RELEASE (nombre)
Libera de la memoria principal el procedimiento (nombre).

Funciones

ACOS (X) - devuelve un valor en coma flotante que es el arco coseno en radianes del argumento X.
X debe ser real y además cumplir ABS(X) 1

-El resultado producido está en el rango:
0 A COS(X) P1

ASIN (X) - igual que ACOS (X), para el arco seno.

OFFSET(X_1,X_2) devuelve un valor offset de un pointer x_1, relativo a un área X_2.
Si X_1 es nulo, devuelve el valor NULL.

PTR (X_1,X_2) devuelve un valor pointer derivado de un valor offset X_1
respecto a un área X_2.

El PL/1 Optimizer impone diferentes requerimientos que el PL1/F, y que son necesarios tener en cuenta a la hora de pasar un programa escrito en PL1/F, para cambiar las instrucciones necesarias, estos son:

1) No puede haber declaraciones ENTRY para procedimientos internos, unicamente para procedimientos externos (en el PL/F eran obligatorias si los tipos argumento/parametros no coincidian)

2) El atributo GENERIC tiene otra sintaxis, que es la siguiente:

```
nombre GENERIC (entry-expresión WHEN (generic-lista descriptores)
               , entry-expresión WHEN (generic-lista descriptores
               ...) ;
```

```
Ejemplo: DECLARE  CALCULO  GENERIC
                   (FIJO  WHEN (FIXED, FIXED),
                   (FLOTANTE  WHEN  (FLOAT, FLOAT),
                   (MIXTO  WHEN  (FIXED, FLOAT)),
```

3) Las funciones preconstruidas que no tienen argumentos deben declararse con el atributo BUILTIN, o a la hora de hacer referencia a las mismas, incluir una lista de argumentos vacia.

```
Ejemplo:  DECLARE   ONCHAR   BUILTIN
                    .
     o bien         .
                    .
     ........ONCHAR ( ) ;
```

4) No existe la función NULLO, se cambia por NULL.

5) También hay cambios en las llamadas al SORT.

Para completar estas referencias es conveniente consultar el manual de definición, que está disponible en el Centro de Cálculo de la Universidad Complutense.