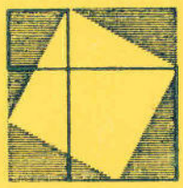
# INFORME TECNICO INTERNO

## Nº. 2

INSTITUTO DE MATEMATICA DE BAHIA BLANCA

INMABB (UNS - CONICET)



UNIVERSIDAD NACIONAL DEL SUR

Avda. ALEM 1253 - 8000 BAHIA BLANCA

República Argentina

I. T. I. N° 2

# MINIMANUAL PARA EL USUARIO DEL SISTEMA VAX 11/780

## Edición de archivos con el editor EDT

## Algunos comandos para el manejo de archivos

Aurora V. GERMANI de POUSA

Susana E. PANZONE

1984

# I EL SISTEMA VAX 11/780

La terminal de video con que cuenta el INMABB está conectada al sistema VAX 11/780.

Este sistema está compuesto por : una "unidad de memoria" (con capacidad de 1.5 Mb, que puede crecer hasta 8 Mb, en palabras de 32 bits) y "elementos periféricos" a saber :

Un "disco" que sirve de memoria auxiliar (con capacidad de 256 Mb)

Una "cinta rápida" (que graba en dos densidades 800 y 1600 bytes/pulgada)

Una "terminal consola" con la que se maneja todo el sistema, ésta es una unidad de entrada y salida de datos y es impresora

Varias "terminales de video", Estas son unidades de entrada y salida de datos, con las que los usuarios se comunican con el sistema

Una "impresora" que es una unidad de salida de datos, con la misma los usuarios pueden obtener las copias de sus programas

Cada usuario tiene asignada una "cuenta" que el sistema individualiza por dos números [n,m]. La cuenta tiene un "nombre" (username) y una "palabra clave" (password) que son necesarias para entrar al sistema.

A cada cuenta se le asigna una parte del disco que se mide en "bloques" (de 512 bytes cada uno). La cuenta "INMABB" tiene asignados 5000 bloques que pueden ser aumentados en 2000 bloques más.

En este disco quedarán grabados los "archivos" (file), nombre genérico de los programas en sus distintas extensiones,

que el usuario "edite" cada vez que haga uso del sistema. Cada archivo queda identificado por un nombre, con éstos el sistema genera un "directorio" para cada cuenta. En realidad los directorios son archivos en los cuales el sistema guarda los nombres de los archivos de una cuenta en orden alfabético.
Este directorio podrá ser consultado por el usuario, pudiendo eliminar del mismo aquellos archivos que crea conveniente.
Los archivos son grabados en el disco por un "editor" que es el nexo entre el usuario y el sistema.

Para procesar los archivos el sistema usa el "lenguaje máquina" que es un lenguaje de "bajo nivel", los usuarios editan los archivos en lenguajes de "alto nivel", para implementar los archivos el sistema tiene "compiladores" que traducen dicho archivo al lenguaje máquina y una vez procesado por el sistema traducen sus resultados al lenguaje que corresponda. Esto hace que el sistema individualize cada archivo además de por el nombre, por su "extensión" que se indica a continuación del nombre por tres letras precedidas por •.
En particular estas letras indicarán el lenguaje que se ha usado para implementar un archivo (•BAS; •FOR; etc. indican que los archivos están codificados en lenguaje BASIC,FORTRAN respectivamente), pero también las características del mismo (•DAT por ejemplo indicará que el archivo es un listado de datos), estos archivos se conocen también con el nombre de "programa fuente".

Al compilar un archivo el sistema genera otra extensión del mismo: • OBJ, dando lugar al "programa objeto".

-Posteriormente el sistema creará el "programa ejecutable" cuya extensión es . EXE, siempre y cuando el programa objeto no contenga errores de sintaxis.
Ahora bien, en general los errores de sintaxis son detectados por el sistema al generar el programa objeto, es decir al compilar el archivo.
Si el usuario implementa su archivo en lenguaje BASIC, el sistema detecta los errores de sintaxis al ingresar cada una de las líneas del mismo. En realidad el compilador BASIC es distinto de los otros compiladores (FORTRAN, COBOL, etc.) y se conoce con el nombre de "intérprete".

La comunicación entre el usuario y el sistema se produce desde una terminal de video, la misma cuenta con un teclado de escritura que permite ingresar caracteres y una pantalla o visor que hace posible leer las instrucciones que se dan al sistema y la respuesta de éste.

Las instrucciones tienen por nombre genérico "comandos". Los mismos se introducen por medio de palabras que describen la función que ejecutan o bien por medio de algunas teclas especiales que reciben el nombre de "llaves" (key).

Cuando se prende la terminal el visor está libre de caracteres y lo único que se observa en la pantalla, en su extremo superior izquierdo, es el cursor que según sea el modelo de la terminal adopta distintas formas: un guión -, o bien un rectángulo ▮ . La función del cursor es indicar el lugar donde aparecerá el caracter que se ingresa por el teclado de escritura, cada vez que se introduce un caracter el cursor se desplazará un espacio a la derecha del mismo. En la pantalla de video pueden ingresarse hasta 80 caracteres y 25 líneas, si se ingresan más líneas que las indica-

das se observa un movimiento en la pantalla, por el mismo desaparecen del visor las líneas superiores.
Como el usuario puede comunicarse con el sistema en forma directa o bien con un editor, esta situación queda evidenciada en la pantalla por símbolos que aparecen a la izquierda del cursor. El hecho que el usuario esté en comunicación con el sistema queda en evidencia con el signo "$"; en el caso que el usuario haya "entrado" al editor (en nuestro caso dicho editor recibe el nombre: EDT) aparecerá en la pantalla el signo "*".

Algunas de las teclas del teclado de escritura permiten el ingreso de comandos al sistema y/o al editor, a continuación se da una descripción del mismo, para que el usuario los reconozca antes de entrar al sistema o bien de editar un programa.

## II DESCRIPCION DEL TECLADO DE LA TERMINAL

En una terminal se pueden observar dos teclados, uno similar al de una máquina de escribir, pero con algunas teclas adicionales, el segundo ubicado a la derecha cuenta con 12 teclas, cada una de las cuales es una "llave" (key) con un significado preciso que el editor asume por defecto (default) es decir en ausencia de otra definición de las mismas.

Hay que notar que este teclado puede tener algunas teclas más, esto depende de la marca y el tipo de la terminal. "Digital" tiene dos tipos de terminales: VT 52 y VT 100, siendo esta última más completa y con otro tipo de configuración en el teclado que la primera.

La terminal asignada al INMABB es una "Data System" que el sistema reconoce como si fuera una VT 52 de "Digital"; en consecuencia se da a continuación la descripción de los comandos que el editor reconoce. Algunos coinciden con los proporcionados por "Data System" en el manual de la terminal, otros difieren de éstos; sin embargo hay que notar que estos comandos pueden ser definidos o programados si el usuario lo cree conveniente.

Comenzamos la descripción de los teclados, indicando en primer término la acción de las llaves del teclado principal.

En el extremo izquierdo se observan las siguientes teclas:

ESC ; TAB ; CTRL ; CAPS LOCKS ; SHIFT

ESC = ESCAPE : Se usa en combinación con letras y definen comandos para el movimiento del cursor o de la pantalla; así tenemos :

ESC/S : Se usa para detener el listado de un programa o detener la "corrida" de un programa.

ESC/Q : Anula la acción del comando ESC/S, permitiendo la continuación del listado del archivo por pantalla o la corrida del programa

Cuando se edita un archivo en modo pantalla :

ESC/A : Permite el movimiento del cursor hacia arriba.

ESC/B : Permite el movimiento del cursor hacia abajo.

ESC/C : Permite el movimiento del cursor hacia la derecha.

ESC/D : Permite el movimiento del cursor hacia la izquierda.

TAB ≡ TABULADOR : Es el tabulador del teclado, presionándolo el cursor se ubica a 7 espacios de donde se encuentra.

CTRL ≡ CONTROL : Como "ESC" se usa en combinación con letras y definen comandos específicos que el editor reconoce, así tenemos :

CTRL/Z : Cuando se edita un archivo en "modo pantalla", presionándolo pasamos a "modo línea".
Presionándolo varias veces se puede salir del editor, reingresando al sistema, pero en este caso el sistema genera un archivo, cuya extensión es: "•JOU".

CTRL/Y ≡ CTRL/C : Permiten salir del editor y reingresar al sistema. Si se los usa durante la edición de un archivo "abortan" su ejecución.
Cancelan un comando entero, no importando las líneas que se hayan entrado.
Se los puede utilizar para "interrumpir" el sistema mientras esté ejecutando el comando en cuestión.
Si se sale del editor usando este comando

también el sistema genera la extensión •JOU del archivo. No es conveniente su uso para salir del editor ya que puede "abortar" la ejecución de un programa.

CTRL/U : Borra la línea que se está ingresando.

CTRL/K : Permite definir llaves y redefinir la acción de las llaves del tablero de comandos.

CTRL/L y CTRL/C : Permite la tabulación por filas y columnas respectivamente.

CAPS LOCKS : Admite dos posiciones, a nivel de las restantes teclas o bien hacia abajo, en un nivel inferior, posición en la que generalmente se encuentra y que hace que la escritura de los caracteres sea en letra mayúscula. Si se desea escribir en letra minúscula basta con presionarlo para que tome otra posición.

SHIFT : Al presionarlo permite la escritura de los caracteres superiores de las teclas dobles. Puede usarse también para ingresar algunos comandos del teclado de llaves. Esta tecla está duplicada a la derecha del teclado.

Las teclas que se observan en el lado derecho del teclado son:

BACK SPACE , LINE FEED , DELETE , RETURN , SHIFT , REPEAT

SHIFT ya fue tratada anteriormente, veamos las acciones de las restantes. Por su importancia describimos en principio :

RETURN : Genera un "car return" o final de línea, al presionarlo el cursor se ubica en el márgen izquierdo de la línea siguiente.

Cada vez que se ingresa un comando al sistema debe

presionárselo, esto quedará evidenciado cuando se dé la descripción de los comandos del sistema, ya que a la derecha de cada uno de ellos figurará el símbolo (R), que indicará su ejecución.
Cuando se edita un programa al terminar cada línea, debe usarse este comando para ingresar a la línea siguiente, en caso de ausencia del mismo, el sistema no reconocerá dicha instrucción y al compilar el programa emitirá un mensaje de error.

BACK SPACE : La acción de esta llave dependerá del modo en que se encuentra el sistema, a saber:
Si estamos en el editor en modo línea presionándola el cursor se moverá hacia la izquierda caracter a caracter permitiéndonos la sobreimpresión de caracteres o bien generar un espacio presionando la barra espaciadora. Lo mismo sucede si nos encontramos fuera del editor.
Si trabajamos en modo pantalla al presionarla el cursor se ubica en el márgen izquierdo de la línea donde se encuentra. Si el cursor ya estuviera en esa posición subirá a la línea siguiente.

LINE FEED : Como la anterior su acción depende del modo en que estamos editando un archivo :
En modo línea, al presionarla el cursor baja a la línea inferior pero manteniendo su ubicación respecto del márgen izquierdo.
En modo pantalla : borra la línea desde donde está ubicado el cursor y hacia la izquierda palabra a palabra.

DELETE : Borra el caracter ubicado a la izquierda del cursor.

REPEAT : Se usa en combinación con otras teclas, manteniéndolo presionado multiplica la función de las restantes.

El teclado principal tiene en su parte superior una línea de teclas, algunas de las cuales pasamos a describir :

OFF LINE : Cuando se la acciona, presionándola, desconecta la terminal del sistema, por lo tanto lo que se escriba en pantalla no quedará registrado en la memoria del mismo. Sin embargo no todos los comandos del teclado responderán.En este caso, para generar un espacio no usaremos RETURN, sino LINE FEED.

ERASE : Permite limpiar la pantalla cuando el comando OFF LINE está accionado.

## III UNA SESION CON LA COMPUTADORA

Una sesión con la computadora comienza cuando el usuario "entra" al sistema desde una terminal de video. Antes de hacerlo debe cerciorarse que no hay otro usuario conecta co al sistema, en el caso en que la terminal esté prendi da, esto queda en evidencia si vemos en la pantalla una última línea con el siguiente mensaje :

"NOMBRE" logged out at "FECHA Y HORA"

En la línea siguiente se verá sólamente el cursor "-" marcando el extremo izquierdo de la misma.
Si la terminal está apagada al prenderla la pantalla estará libre de caracteres y lo único que se observa es en el extremo superior de la misma el cursor.
Se entra al sistema en cualquiera de los dos casos presionando "RETURN". Cada vez que se ingresa un comando deberá presionarse esta tecla, indicaremos esto con el símbolo (R).
La respuesta del sistema al comando "RETURN" (R) es la leyenda :

Username : -

quedando el cursor a la espera del nombre de la cuenta.
Se ingresa éste y el sistema preguntará por la "palabra clave" que tiene esa cuenta; la situación en el visor será entonces

Username : INMABB (R)

Password : (R)

Como "Password" es una palabra clave y sólamente la cono

ce el usuario, al ingresarla no aparece en la pantalla.
Si el usuario no ingresa la palabra clave correcta, el sistema imprime un mensaje de "error" y se debe reiniciar el proceso de entrada al sistema, presionando nuevamente "RETURN".
Si la clave es correcta aparece en el extremo izquierdo de la línea siguiente el signo "$" y a continuación el cursor a la espera de instrucciones.
En la pantalla tendremos entonces

```
Username : INMABB

Password :

$ -
```

Las instrucciones que se dan al sistema se conocen como "comandos". Estos son palabras que describen la función que ejecutan, se escriben a continuación del signo "$" y se ingresan al sistema presionando "RETURN".
Si el comando es correcto el sistema lo ejecuta, en caso contrario imprime un "mensaje de error" en la pantalla.

Suponiendo que se quiera ingresar el comando "COPY" pero se haya ingresado éste con un error en su escritura, se tiene la siguiente situación :

```
$ CAPY                    (R)

% DCL - W - IWERB : unrecognized command

  \CAPY\
```

DCL : Indica que el error proviene del intérprete de comandos (= DCL).

W : Es la abreviatura de "warning".

IWERB : Es el nemotécnico del mensaje.

Si al escribir el comando que se quiere ingresar se cometen errores de escritura éstos se pueden corregir recordando que:

"DELETE" borra los caracteres de derecha a izquierda.

"CTRL/U" borra la línea que se está escribiendo, ejecuta un "RETURN" y permite reingresar otra línea.

"CTRL/C" ≡ "CTRL/Y" cancelará un comando entero no importando las líneas que se hayan entrado, pero interrumpe el sistema mientras está ejecutando otro comando.

Los comandos en general podrán ser abreviados, ya que el sistema los reconoce por sus primeros cuatro caracteres, pero también pueden usarse menos caracteres.
Así los comandos "SET" y "SHOW" son reconocidos por sus dos primeras letras : "SE" y "SH" respectivamente.
El comando "DIRECTORY" será reconocido por "DIR".

Al entrar un comando no hace falta especificar todos los parámetros necesarios, ya que el sistema preguntará por la información adicional que le permita su ejecución, es decir si en un comando estuviera involucrado, por ejemplo, el nombre de un archivo en caso de no especificarse el mismo, el sistema emite un mensaje.
Así si se desea imprimir un archivo el comando que se usará es "PRINT" y a continuación el nombre de dicho archivo :

```
$ PRINT   NOMBRE          (R)
```

Pero también se puede introducir el comando de la siguiente forma

```
$ PRINT                   (R)
```

El sistema imprime :

```
$ - File : NOMBRE         (R)
```

Para terminar la sesión con la computadora el usuario debe despedirse del sistema.
Usando el comando

```
$ LOGOUT                (R)
```

el sistema responde imprimiendo en la pantalla el mensaje

```
INMABB Logged out at 18-OCT-1983  17:35 ; 10-38.
-
```

El cursor aparece al principio de la línea siguiente, en ese momento puede entrar al sistema otro usuario o bien apagar la terminal.

El comando "LOGOUT" puede abreviarse, basta con poner :

```
$ LOGO                  (R)
```

o simplemente

```
$ LO                    (R)
```

para despedirse del sistema operativo.

NOTA: En caso de necesitar ayuda al introducir un comando y no disponer del manual del sistema use el comando "HELP"

```
$ HELP                  (R)
```

El sistema le muestra por pantalla un listado de todos los comandos, pudiéndose pedir información entonces de cualquiera de los que aparecen en la lista.
Si se pide

```
$ HELP PRINT            (R)
```

la información que aparece en la pantalla es un resúmen del comando "PRINT" a saber : los parámetros que se requieren para su ejecución, y el "DEFAULT" del mismo, es decir lo que el sistema asume por defecto.

## IV EDICION DE ARCHIVOS CON EL EDITOR : EDT.

Una vez que hemos entrado en la cuenta, podemos desear editar un archivo o programa; para esto debemos en primer término crear el mismo para lo cual debemos llamar al editor y posteriormente indicar al sistema el nombre y extensión del archivo en cuestión.

Con el comando :

```
$ EDIT/EDT                    (R)
```

le indicamos al sistema que deseamos entrar al editor, la respuesta de aquel es un mensaje donde pregunta el nombre del archivo a editar :

```
$ File :
```

Nuestra respuesta es escribir en la pantalla el nombre y extensión del archivo oprimiendo a continuación el comando "RETURN".
Como el archivo que nombramos no fue editado con anterioridad el sistema manda un mensaje por pantalla :

```
Input file does not exist
[EOB]
* -
```

Con el mismo nos informa en primer têrmino que el archivo no estaba registrado en el directorio de la cuenta. En segundo término la leyenda [EOB] nos indica que creó un archivo con el nombre pedido. ([EOB] significa final de archivo, veremos después cómo funciona). En la tercera línea aparece el asterisco "*" y el cursor "-" que nos indica que el editor está a la espera de comandos.

Resumiendo hasta este momento en la pantalla tenemos :

```
$ EDIT/EDT                          (R)
$ File : NOMBRE.EXT                 (R)
$ Input file does not exist
[ EOB]
* -
```

También se puede entrar al editor de la siguiente manera :

```
$ EDT NOMBRE.EXT                    (R)
```

La respuesta del sistema será la misma.

Los mismos pasos deberemos dar en caso de querer llamar un archivo que ya fue editado, para su consulta, corrección o ejecución, en este caso la respuesta del sistema es distinta y en pantalla aparecerá la primera línea del archivo o programa que pedimos, de manera que ahora tendremos :

```
$ EDIT/EDT                          (R)
$ File : NOMBRE.EXT                 (R)
1 Primera línea del texto del archivo.
* -
```

Cada vez que aparece "*" el editor indica que está listo para ingresar comandos.
Hay dos formas de editar o consultar un archivo que se conocen bajo el nombre de "modo pantalla" y "modo línea". Veremos a continuación el modo pantalla.

## V EDICION DE ARCHIVOS EN MODO PANTALLA

Cuando se edita un archivo en modo pantalla se pueden ingresar hasta 80 caracteres por línea, el ingreso de los mismos se hace directamente de acuerdo a la sintaxis del lenguaje usado para la creación del mismo.

Una vez que hemos creado el archivo en el editor, le daremos a éste el comando "CHANGE" al cual el sistema reconoce por su primera letra; estamos entonces en la siguiente situación

```
$ EDT NOMBRE.EXT                    (R)

  Input file does not exist

  [EOB]

* C                                  (R)
```

El editor responde a este comando, borrando la pantalla y ubicando el cursor y el final de archivo [EOB] en la esquina superior izquierda de la pantalla, donde se comenzará la escritura del archivo.

Al comenzar con la misma el final de archivo [EOB] bajará a la línea siguiente manteniéndose siempre en el márgen izquierdo.

Cada vez que se termina una línea ¡debe oprimirse el comando "RETURN"!

De esta manera el editor reconocerá el final de línea..

Al accionar "RETURN" el cursor se ubica en la línea siguiente en el margen izquierdo.

Cuando se edita un archivo en modo pantalla "por defecto" el editor acepta los comandos que se introducen usando las llaves del teclado ubicado a la derecha del teclado principal, por lo tanto para concluir con la edición del archivo y salir del editor usaremos la llave que nos permite la inserción de

comandos :

AZUL/7

y al mensaje del editor :

Command :

responderemos escribiendo EXIT y presionando la llave "ENTER". La respuesta a este comando es la aparición en pantalla del signo '$' lo que nos indica que hemos salido del editor y el sistema está a la espera de nuevos comandos.
La situación en la pantalla será :

ULTIMA LINEA DEL TEXTO :

[EOB]

Command : EXIT

$

El archivo creado quedará grabado y su nombre y extensión figurará en el directorio con el número de versión 1, ya que recién fue creado.
Si en vez del comando "EXIT" se hubiera usado "QUIT" para salir del editor, el archivo no quedará grabado, por lo tanto su nombre no quedará en el directorio.

## VI CORRECCION DE ARCHIVOS

Una vez editado un archivo puede ser necesaria su corrección Para efectuar la misma se lo puede hacer de dos maneras, haciendo uso del teclado de comandos o modo "keypad" al cual el editor asume por defecto o bien en modo "nokeypad" en el cual los comandos se introducen por combinaciones de las letras y números del teclado principal de la terminal, si este es el caso una vez llamado el archivo por medio del editor, deberemos cambiar el "modo" al mismo por medio del comando "SET NOKEYPAD", tendremos entonces :

```
$ EDT  NOMBRE.EXT                    (R)
  1  PRIMERA LINEA DEL TEXTO DEL ARCHIVO
  * SET NOKEYPAD                     (R)
  * C                                (R)
```

El comando "C = CHANGE" produce en la pantalla la aparición de la totalidad del archivo o bien de la primera parte del mismo quedando el cursor marcando el primer caracter en la parte superior izquierda de la pantalla a la espera de los comandos.

Si solamente se usan los comandos

```
$ EDT  NOMBRE.EXT                    (R)
  1  PRIMERA LINEA DEL TEXTO DEL ARCHIVO
  * C                                (R)
```

Al no usar el comando "SET NOKEYPAD" (con el cual se anula el teclado de llaves para introducir comandos) la corrección se puede hacer en forma directa, una vez terminada la misma saldremos del editor introduciendo el comando "EXIT" el cual

hará generar la nueva versión del archivo y esta aparecerá en el directorio con el número de la versión aumentado en una unidad respecto de la versión anterior. Recordamos que en este caso los comandos se introducen usando : AZUL/7. De esta manera pedirá el comando a introducir :

```
Command : EXIT              (ENTER)
$
```

El comando "SET NOKEYPAD" puede introducirse también de la misma forma que el comando "EXIT"; si hacemos : AZUL/7 e introducimos este comando como respuesta al editor :

```
Command : SET NOKEYPAD     (ENTER)
```

A partir de este momento el editor desconocerá las funciones que se introduzcan haciendo uso de las llaves del teclado de la derecha.

## VI . 1 MODO KEYPAD

Para el uso del editor en modo pantalla ( en modo "keypad", que es el que el editor asume en ausencia de otra información) se usan las teclas azul,roja y blanca y el teclado de la derecha.

AZUL : Se usa para definir funciones o comandos combinándola con el teclado de la derecha. Pero también admite combinaciones usando los números del teclado principal como veremos después.

ROJO ≡ HELP : Cuando se la acciona aparece en pantalla la descripción de los comandos del teclado de llaves. Permite su combinación con éstas, para conocer su acción.

GRIS : Borra a la derecha del cursor (incluyendo el "CAR RETURN". No permite ingresar un nuevo texto.

AZUL / GRIS : Devuelve la última línea borrada, al accionar la llave gris.

Si se quiere borrar una línea usando la llave "gris" el cursor ¡debe ubicarse al principio de la misma!

Cuando en el editor se usa el "modo keypad" este asume por defecto el "modo advance". Esto significa que los movimientos del cursor serán de izquierda a derecha en cada línea y de arriba hacia abajo cuando recorra el texto. Si los movimientos del cursor son los opuestos a los anteriores diremos que el editor se encuentra en "modo backup".

El cambio de "modo advance" a "modo backup" se obtiene

presionando la llave número 5.

El cambio de "modo backup" a "modo advance" se logra presionando la llave número 4.

A continuación damos una descripción de la acción que genera cada una de las llaves, observando que los movimientos del cursor serán de acuerdo al modo en que se encuentra el editor.

0 : Se recorren los principios de línea.

1: El cursor recorre las líneas ubicándose en la primera letra de cada palabra.

2: Se recorren los finales de línea

3: Marca el final del "selector de rango" (se explicará después). Produce un "corte" en el archivo.

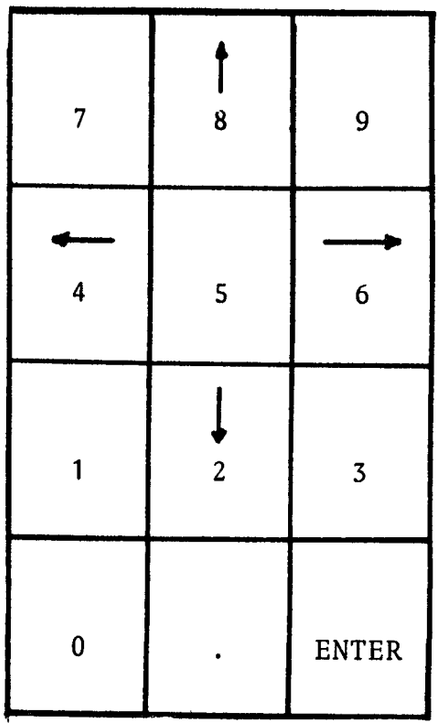
6: Borra el caracter que marca el cursor.

7: Recorre un archivo página a página.

8: Es el comando de "búsqueda" (se explicará después)

9: Borra la palabra que marca el cursor.

.: Activa el "selector de rango" (se explicará después)



Accionando las llaves después de accionar el comando "AZUL"

AZUL / 0 : Genera espacio a la derecha del cursor, permitiéndonos ingresar caracteres : en la línea superior si el cursor está en el márgen izquierdo, en la misma línea si el cursor se encuentra en el interior o en el final de la línea (en este caso los caracteres ubicados a la derecha del cursor son desplazados a la línea de abajo).

AZUL / 1 : Cambia de mayúscula a minúscula y viceversa la letra que marca el cursor.

AZUL / 3 : Permite "pegar" la parte del archivo que se "cortó" al accionar la llave 3 (Este proceso lo describiremos con más detalles).

AZUL / 4 : Ubica el cursor al final del archivo.

AZUL / 5 : Ubica el cursor al principio del archivo.

AZUL / 6 : Anula la acción de la llave 6, devolviendo el "último" caracter que se borró al accionar la llave 6.

AZUL / 7 : Permite introducir comandos, la respuesta del editor es la impresión del mensaje :
Command.
A continuación escribiremos el comando que queremos usar. Los mismos serán ejecutados al presionar la llave "ENTER". Esto nos permite por ejemplo salir del editor ya sea con el comando "EXIT" o bien con el comando "QUIT" y entrar nuevamente al sistema.
También podemos anular las funciones del teclado de comandos, de esta manera el editor dejará de asumir por defecto el "modo keypad". El comando

que produce este cambio es : "SET NOKEYPAD".
Al presionar "ENTER" los comandos deberán introducirse haciendo uso del teclado principal.

AZUL / 8 : Permite la "búsqueda" de una cadena de caracteres (string) en el archivo, al accionarlo el editor imprime en la parte inferior de la pantalla :
Search for :
A continuación se escribe el caracter o palabra que deseamos, introduciéndola con el comando "ENTER". Presionando la llave número 8 el cursor se ubicará señalando el caracter o palabra que se pidió. La búsqueda se hará en "modo advance" o "modo backup", si no encuentra el caracter pedido imprime el mensaje
String was not found.

AZUL / 9 : Anula la acción de la llave 9. Devuelve la "última" palabra que se borró al accionar 9.

Las llaves 2,4,6,8 tienen en la parte superior flechas orientadas hacia abajo, a la izquierda, a la derecha y hacia arriba respectivamente, lo que indica los posibles movimientos del cursor desde el lugar donde está ubicado y en la dirección que indica dicha flecha. Estos movimientos se lograrán combinando la tecla "SHIFT" con cada una de estas llaves; resumiendo:

SHIFT / 2 El cursor se ubica marcando el caracter de la línea de abajo.

SHIFT / 4 El cursor se ubica en el caracter que está a su izquierda.

SHIFT / 6 El cursor se ubica en el caracter que está a su derecha.

SHIFT / 8 : El cursor se ubica marcando el caracter de la línea de arriba.

Estos mismos movimientos del cursor se pueden lograr usando combinaciones del comando "ESCAPE" con letras

ESC / A ≡ SHIFT / 8

ESC / B ≡ SHIFT / 2

ESC / C ≡ SHIFT / 6

ESC / D ≡ SHIFT / 4

Las funciones descriptas anteriormente y que se introducen con las llaves que tienen los números : 0,1,2,3,4,5,6,7, y 9 así como la acción de la tecla gris y las combinaciones de éstas con la tecla azul pueden ser multiplicadas introduciendo números (después de presionar la tecla azul) con el teclado principal. Este número aparecerá impreso en la parte inferior de la pantalla; la acción ahora será la siguiente :

Presionando

AZUL /n/ 0 El cursor se ubicará a n filas de donde estaba.

AZUL /n/ 1 El cursor se ubicará a n palabras de donde estaba.

AZUL /n/ 2 El cursor se ubicará indicando el final de línea a n filas de donde estaba.

AZUL /n/ 6 Se borrarán n caracteres hacia la derecha del cursor.

AZUL /n/ 7 Se saltan n páginas.

AZUL /n/ 9 Se borrarán n palabras.

Etc.

EL USO DEL SELECTOR DE RANGO

1 . CORTE Y PEGADO (CUT-PASTE)

Cuando se acciona la llave marcada con · se activa el "selector de rango", esto nos permite "cortar" parte del archivo y "pegarla" posteriormente en otro lugar del mismo que hemos elegido.

La parte del archivo que queremos insertar en otro lugar del mismo la seleccionamos recorriendo sus líneas, palabras o caracteres con el cursor una vez que se ha accionado la llave ·, el proceso de selección termina accionando la llave 3, cuya acción es el "corte" (cut) de la parte del archivo elegida; la cual desaparece del visor, permitiendo el translado de la misma al lugar elegido, el cual queda determinado por el cursor (que será llevado a ese lugar), accionando la llave AZUL/3 efectuamos el "pegado" (paste) del texto seleccionado.

2 . CAMBIO DE LETRAS

Accionada la llave ·, recorremos parte del archivo con el cursor, accionando AZUL/1, en la parte del archivo recorrida las letras mayúsculas se transformarán en minúsculas y viceversa.

Solamente de estas dos maneras se termina con la selección de rango.

Ahora bien si el "selector de rango" fue activado, al presionar nuevamente la llave · el editor emite el mensaje

Select range is already active.

NOTA FINAL : Puede ocurrir que los comandos no realicen las funciones detalladas anteriormente; en ese caso "APAGUE LA TERMINAL". Al volver a prenderla esta estará nuevamente en

las condiciones requeridas, para el correcto funcionamiento de los comandos. También puede desconectarla presionando la llave "OFF LINE".

OBSERVACION : La descripción de estos comandos se encuentra sintetizada en el cuadro que se encuentra en la pared, arriba de la terminal.

| LLAVE | ACCION | COMBINADA CON AZUL | SHIFT |
|---|---|---|---|
| 0 | Recorrer los principios de línea | Abrir línea | ——— |
| 1 | Recorrer línea palabra a palabra | Cambiar letra mayúscula por minúscula y viceversa | ——— |
| 2 | Recorrer finales de línea | Borrar línea (⟶) | ↓ ESC/B |
| 3 | Final de selección de rango (corte) | Ubicar texto en otro lugar (pegado) | |
| 4 | Modo advance | Final de archivo | ← ESC/D |
| 5 | Modo back up | Principio de archivo | ——— |
| 6 | Borrar el caracter que marca el cursor | Devuelve el último caracter borrado | → ESC/C |
| 7 | Recorrer el archivo página a página | Introducción de comandos | ——— |
| 8 | Búsqueda de caracteres | Permite la búsqueda de palabras o caracteres | ↑ ESC/A |
| 9 | Borrar la palabra que marca el cursor | Devuelve la última palabra borrada | ——— |
| · | Activar el selector de rango | ——— | ——— |
| ENTER | Entrar comandos | ——— | ——— |
| GRIS | Borrar ⟶(incluído el final de línea) | Anula la acción de la tecla "GRIS" y también de AZUL/2 devolviendo la última línea borrada | |

ESC/S : Detener la corrida por pantalla

ESC/Q : Anular la acción de ESC/S

CTRL/Y ≡ CTRL/C : Salir del editor

CTRL/Z ≡ Anular la acción de un comando dado al sistema (se puede salir del editor)

## VI . 2 MODO NOKEYPAD

Para mover el cursor en este caso deberemos tener en cuenta que los comandos se introducirán con una combinación de caracteres que involucran números y letras según la convención siguiente :

L indica LINEA
C indica CARACTER
W indica PALABRA (WORD)
D se usa para borrar (DELETE)
I se usa para insertar
S se usa para sustituir

Ahora bien para que el cursor se ubique en el lugar deseado las tres primeras letras deberán ser precedidas por un número, el comando se introducirá presionando "RETURN", tendremos entonces :

| | | |
|---|---|---|
| nL | (R) | El cursor baja n líneas |
| nC | (R) | El cursor avanza n caracteres a la derecha |
| nW | (R) | El cursor avanza n palabras a la derecha |

Los movimientos hacia "arriba" y a la "izquierda" se indicarán con el signo "-" precediendo los comandos anteriores.

| | | |
|---|---|---|
| -nL | (R) | El cursor sube n líneas |
| -nC | (R) | El cursor se ubica n caracteres a la izquierda |
| -nW | (R) | El cursor se ubica n palabras a la izquierda |

Para borrar líneas, palabras o caracteres usaremos la letra D, a saber

| | | |
|---|---|---|
| D1C | (R) | Borra el caracter que marca el cursor |
| DnC | (R) | Borra n caracteres |

Ubicado el cursor en la primera letra de una palabra, con el comando

| | | |
|---|---|---|
| D1W | (R) | Borra la palabra que indica el cursor |
| DNW | (R) | Borra n palabras |

Ubicando el cursor en el principio de la línea, el comando

| | | |
|---|---|---|
| D1L | (R) | Borra la línea que indica el cursor |
| DnL | (R) | Borra n líneas incluyendo la que indica el cursor |

Si se desea cambiar un caracter por otro, primero se lo borra (D1C (R)) y después se inserta el nuevo caracter, usando antes de presionar "RETURN" el comando "CTRL/Z" :

I (nuevo caracter) (CTRL/Z) (R).

Para crear espacio entre dos caracteres, usaremos el mismo comando pero en vez de introducir caracteres presionaremos la barra espaciadora la cantidad de veces que sea necesario según los espacios a crear, tendremos entonces

I (espacio) CTRL/Z (R)

El reemplazo de una palabra por otra se hará usando el comando

S/ (palabra) / (palabra nueva) / (R)

El editor comienza la búsqueda de la palabra en "modo advance" por lo tanto puede impimir un mensaje

String was not found

En caso que la misma esté ubicada algunas líneas arriba de donde está ubicado el cursor o bien a su izquierda, en este caso usare

mos el comando anterior precedido por el signo "-"

-S / (palabra) / (palabra nueva) / (R)

Podemos cambiar palabras de lugar de la siguiente manera: ubicado el cursor en la primera letra de la misma, la borramos :

D1W (R)

A continuación ubicamos el cursor donde debe quedar insertada la palabra y damos el comando :

UNDW (R)

El proceso de "corte" de parte del archivo, y "pegado" en otra parte del mismo se puede hacer ahora, introduciendo los comandos "CUT" y "PASTE" de la siguiente manera: ubicado el cursor en la primera línea de la parte que se quiere "cortar" con el comando:

nL CUT = SHOW (R) Del visor desaparecen las n líneas que deben ser transladadas

Con el comando :

mL PASTE = SHOW (R) La parte del texto cortada quedará ubicada m líneas abajo

Con el comando :

-mL PASTE = SHOW (R) La parte de texto cortada quedará ubicada m líneas arriba

Una vez terminada la corrección se da el comando

EXIT (R)

Aparece en la pantalla el * que nos indica que el editor está a la espera de comandos, introducimos el comando "SET KEYPAD" y luego nuevamente "EXIT" con lo cual la sesión terminará; y saldremos del editor. La situación en pantalla será entonces

* SET KEYPAD (R)

* EXIT (R)

$

## VII. DESARROLLO DE PROGRAMAS

Son cuatro los pasos a seguir en el desarrollo de un programa:

1) CREACION DEL PROGRAMA FUENTE

2) COMPILACION DEL PROGRAMA FUENTE PARA OBTENER EL PROGRAMA OBJETO (.OBJ)

3) "LINKEADO", ENCADENAMIENTO O ARMADO DE LA TAREA, A PARTIR DEL PROGRAMA OBJETO PARA OBTENER EL PROGRAMA EJECUTABLE (.EXE)

4) CORRECCION Y EJECUCION DEL PROGRAMA

### 1) CREACION DEL PROGRAMA FUENTE

Usando el editor se crea el archivo que contiene las instrucciones del mismo en el formato del lenguaje que se usará.

### 2) COMPILACION DEL PROGRAMA FUENTE

| Comando : | Invoca : |
|---|---|
| $ BASIC ≡ BAS | VAX-11 BASIC Compiler |
| $ BLISS | VAX-11 BLISS-32 Compiler |
| $ COBOL / C 74 | VAX-11 COBOL-74 Compiler |
| $ FORTRAN ≡ FOR | VAX-11 FORTRAN Compiler |
| $ MACRO | VAX-11 MACRO Assembler |
| $ PASCAL | VAX-11 PASCAL Compiler |

Al utilizar uno de estos comandos se invoca al compilador el cual "chequea" el programa fuente, es decir lo examina o verifica, detectando errores de sintaxis si los hubiera, en este caso emite un mensaje indicándolos, debiendo procederse a la corrección del mismo antes de continuar el proceso.
Si no hay errores de programación y/o de sintaxis, traduce el

programa fuente al lenguaje máquina, dando lugar al"programa objeto".

3) "LINKEO" DEL PROGRAMA OBJETO

El programa objeto no es ejecutable por la computadora pues puede tener llamadas a otros programas o rutinas que tienen que ser combinadas con el programa objeto antes de ser ejecutado, esta es la función del comando "LINK"

$ LINK (R)

4) EJECUCION DE UN PROGRAMA

El comando "RUN" ejecuta el programa "linkeado" en el paso anterior o sea el programa ejecutable (.EXE)

$ RUN (R)

## VII . EL COMPILADOR BASIC

Hemos visto al principio que el compilador BASIC es un intérprete, entramos al mismo usando el comando

```
$ BASIC                    (R)
```

A partir de este momento podemos usar el sistema como una calculadora para lo cual solamente hay que usar el comando "PRINT" y tener en cuenta la forma de codificación de las operaciones aritméticas :

| OPERACION | CODIGO |
|---|---|
| Suma | + |
| Resta | - |
| Multiplicación | * |
| División | / |
| Potenciación | ^ (eventualmente ** ó ↑) |

Además el orden de prioridad que sigue el sistema es el siguiente :

1°) Potencias
2°) Multiplicaciones y/o divisiones
3°) Sumas y/o restas

Si se hacen uso de parêntesis, operará de "adentro hacia afuera"
Al usar el comando

```
$ BASIC                    (R)
```

la respuesta del sistema es

```
VAX-11 BASIC V1.Ø
Ready
-
```

El cursor queda a la espera de instrucciones.
Con el comando

```
PRINT  2*3                 (R)
```

el sistema responde

```
6
Ready
-
```

Se puede pedir también

```
PRINT (2+10-8)^ 5                    (R)
```

respondiendo el sistema :

```
64
Ready
-
```

Para salir del compilador y reingresar al sistema se debe presionar CTRL/Y ≡ CTRL/C.

# VIII ALGUNOS COMANDOS PARA EL MANEJO DE ARCHIVOS

## VIII 1 IDENTIFICACION DE ARCHIVOS

Los archivos se identifican por :

i) Lugar físico donde son almacenados por el sistema (Discos,cintas magnéticas, disquetes, etc.)

ii) Directorio que es el archivo que genera el sistema. En él se encuentran los nombres de los distintos archivos que pertenecen a una determinada cuenta.

iii) Nombre que es un conjunto de 1 a 9 caracteres.

iv) Tipo o extensión que indica el lenguaje utilizado o bien el tipo de archivo, está compuesto de 0 a 3 caracteres precedido por . .

v) Versión que es un número entero cuyo rango oscila entre 1 a 32767 y que el sistema asigna a un archivo cuando es creado o corregido o bien pedido desde el editor.
En el directorio podrá aparecer un archivo con distintas versiones.
Cuando se pide un archivo, si no se pide una versión determinada por default el sistema asume la última.

En nuestro caso el lugar físico donde se almacenan los archivos es el disco del sistema que se identifica por las siglas "_DRAØ" Al no haber otra posibilidad de almacenamiento no es necesario identificar los archivos con la sigla en cuestión, ya que el sistema lo asume por defecto.

Suponiendo que estamos trabajando en la cuenta : INMABB con el comando "DIRECTORY" ≡ "DIR" se obtiene un listado de los archivos almacenados en dicha cuenta en orden alfabético.

Comando :

```
$ DIR                                   (R)
```

El sistema responde :

```
Directory _DRAØ : [ INMABB]

AAA.DAT; 2   COMANDO 1.;6    COMANDO 1.;5
PROGRAMA. BAS;3    PROGRAMA.EXE;1    PROGRAMA.OBJ;1

Total of 6 files
```

El mensaje final indica la cantidad total de archivos.
Observamos que cuando hay varias versiones de un mismo programa están ordenadas de mayor a menor.

## VIII.2 INFORMACION ESPECIFICA DE LOS ARCHIVOS

Hemos dicho que con el comando "DIR" se obtiene el listado completo de los archivos de una cuenta, pero también se puede usar para conocer cuántas versiones hay en el directorio de un archivo de un determinado "Nombre" y extensión: ".EXT"; así usaremos

```
$ DIR  NOMBRE  .EXT; *                (R)
```

Respuesta del sistema :

```
Directory _DRAØ : [INMABB]
NOMBRE .EXT; n1   NOMBRE.EXT; n2   NOMBRE.EXT; n3
Total of 3 files.
```

También podemos usar

```
$ DIR NOMBRE.*;*                      (R)
```

La respuesta del sistema será un listado de todas las versiones de cada una de las extensiones del archivo cuyo nombre se indicó.

El uso del asterisco "*" que acompaña al comando se usará para indicarle al sistema que se desea toda la información que posee respecto del lugar en el cual está ubicado. En el primer caso "*" ocupa el lugar del número de versión; en el segundo caso además ocupa el lugar de la extensión en la forma: ".*".

Si se da el comando :

```
$ DIR *.EXT; *                        (R)
```

El sistema da un listado de todas las versiones de cada uno de los archivos que tienen una determinada extensión.

Para obtener más información sobre los archivos que hay en una cuenta, se usará el comando :

```
$ DIR/FULL                    (R)
```

El sistema responde con un listado de cada uno de los archivos indicando sus características, la protección de cada uno de ellos y la cantidad de espacio (blocks) que ocupa en el disco. Por último imprime un mensaje donde indica la cantidad de bloques utilizados, así como la cantidad de bloques libres con que dispone la cuenta.

Esto último también se puede obtener con el comando :

```
$ SHOW QUOTA                  (R)
```

## VIII.3 SUBDIRECTORIOS

Normalmente cada usuario tiene acceso a una sola cuenta; en consecuencia hay un directorio por cada usuario y en él se deben mantener archivos de todo tipo. Es conveniente entonces tener en el directorio archivos especiales que sean capaces de contener otros archivos, estos se llaman "subdirectorios" y tienen una extensión especial: .DIR, lo que les da características especiales, en particular los subdirectorios tienen distinta "protección" que la de los archivos comunes.

Debemos tener en cuenta que un archivo o programa se crea llamando al editor.

Para crear subdirectorios se usará el comando : "CREATE/DIRECTORY".

Si en la cuenta: INMABB se desea crear un subdirectorio cuyo nombre sea: SBD.DIR, procedemos de la siguiente manera.

```
$ CREATE/DIRECTORY                    (R)
```

El sistema pregunta

```
$ File : [ INMABB.SBD]                (R)
```

El directorio de la cuenta INMABB constará ahora de los siguientes archivos :

```
AAA.DAT;2    COMANDO 1.;6    COMANDO 1.;5
PROGRAMA.BAS;3    PROGRAMA.EXE;1    PROGRAMA.OBJ;1
SBD.DIR;1
```

Ahora bien el manejo del subdirectorio, es distinto del de los otros archivos, ya que se usa como una cuenta especial (dentro de la cuenta INMABB) bastando para esto el cambio del default del sistema.

El comando "SHOW DEFAULT" nos permite conocer qué es lo que

asume el sistema, en nuestro caso tendremos

```
$ SHOW DEFAULT                    (R)
```

Respuesta del sistema :

```
_DRAØ : [ INMABB]
```

Esto significa que al llamar al editor todo programa pasará al directorio general del INMABB.
Para entrar en el subdirectorio y poder editar archivos hay que cambiar el default del sistema usando el comando

```
$ SET DEFAULT  [ INMABB.SBD]      (R)
```

Con el comando

```
$ SHOW DEFAULT                    (R)
```

la respuesta será :

```
_DRAØ : [ INMABB.SBD]
$
```

lo que nos permite trabajar en el archivo SBD.DIR que fué creado anteriormente. Los programas que ahora se editen quedarán en este subdirectorio y no figurarán en el directorio de la cuenta.

Al comando "DIR" el sistema responderá con el listado de los archivos del subdirectorio

```
$ DIR                             (R)
Directory _DRAØ : [ INMABB.SBD]
EV.FOR;1      LER.FOR;1
Total of 2 files.
```

El mantenimiento y uso de los subdirectorios es similar al del directorio general, pudiéndose crear subdirectorios del subdirectorio, a los mismos se accederá cambiando nuevamente el default del sistema. Así si en el subdirectorio SBD.DIR se desea tener el subdirectorio: BIBL.DIR lo creamos de la misma manera

que el primero

```
$ CREATE/DIRECTORY                    (R)
File : [SBD.BIBL]                     (R)
```

y entonces en el directorio [INMABB.SBD] tendremos.

```
EV.FOR;1    LER.FOR;1    BIBL.DIR;1
```

Recordamos que para hacer uso del subdirectorio BIBL.DIR deberemos cambiar el default: del subdirectorio SBD.DIR al subdirectorio BIBL.DIR; con el comando

```
$ SET DEFAULT  [SBD.BIBL]             (R)
```

Entramos en el subdirectorio BIBL.DIR, así si pidiéramos

```
$ SHOW DEFAULT                        (R)
```

el sistema responde

```
_DRAØ:  [INMABB.SBD.BIBL]
```

Para volver a la cuenta INMABB, deberá cambiarse el default con el comando

```
$ SET DEFAULT [INMABB]                (R)
```

Al subdirectorio BIBL.DIR se puede acceder directamente desde el directorio general con el comando SET DEFAULT, pero indicando al sistema que se encuentra en el subdirectorio SBD.DIR :

```
$ SET DEFAULT  [INMABB.SBD.BIBL]      (R)
```

El proceso podrá repetirse a partir del subdirectorio BIBL.DIR. De esta manera el directorio tendrá distintos niveles

```
NIVEL SUPERIOR :
SEGUNDO NIVEL   :   [INMABB.SBD]
TERCER NIVEL    :   [INMABB.SBD.BIBL]
```

## VIII.4 CONSULTA DE ARCHIVOS

Con el comando "TYPE" el sistema muestra el archivo pedido en la pantalla

$ TYPE NOMBRE.EXT (R)

En la pantalla se vuelca toda la información que contiene la última versión del archivo pedido.

El texto puede ser interrumpido usando combinaciones del comando "CTRL" de la terminal a saber :

CTRL/S : Suspende la salida del listado y la ejecución de un comando

CTRL/Q : Permite continuar con el listado y la ejecución del comando interrumpido con CTRL/S.

CTRL/C = CTRL/Y : Abortan la salida o ejecución de un comando permitiendo el ingreso de otro.
Al accionar CTRL/C o CTRL/Y aparece en la pantalla el signo $ lo que indica que el sistema está a la espera de nuevas órdenes.

## VIII.5 MANTENIMIENTO DEL DIRECTORIO

Cada vez que se edita un archivo y se sale del editor con el comando "EXIT" el sistema crea una nueva versión del mismo, de esta manera en el directorio pueden aparecer varias versiones de las distintas extensiones. En general es suficiente la última versión de un archivo para su consulta y/o ejecución, las restantes ocuparán innecesariamente lugar en el disco; lo mismo sucede cada vez que se compila y ejecuta un programa.

Es conveniente entonces eliminar del disco aquellos archivos excedentes, borrándolos, para esta operación se usan los comandos : "DELETE" y "PURGE".

Con el comando :

```
$ DELETE NOMBRE.EXT; n                (R)
```

o bien :

```
$ DELETE                              (R)
$ File : NOMBRE.EXT; n                (R)
```

el sistema borra la versión n del archivo cuyo nombre y extensión se indican, manteniendo las otras versiones.

Con este comando puede usarse "*" como comodín, lo cual da lugar a las siguientes posibilidades de uso de este comando

```
$ DELETE                              (R)
$ File: NOMBRE.EXT; *                 (R)
```

elimina todas las versiones con ese nombre y extensión.

```
$ DELETE                              (R)
$ File : NOMBRE.*;*                   (R)
```

elimina el archivo cuyo nombre se indica cualquiera sea su extensión o número de versión :

```
$ DELETE                              (R)
$ File : *, EXT; *                    (R)
```

Elimina todas las versiones de los posibles archivos que tengan la extensión que se indica.

```
$ DELETE *.*;*                    (R)
```

Borra todos los programas !!!

Si se desean conservar varias versiones de un archivo puede usarse otra forma del comando "DELETE", este es el comando "DELETE/CONFIRM" que puede abreviarse : "DEL/CON".
El sistema ejecuta este comando preguntando si debe borrar o no las distintas versiones del archivo que se indica; en caso afirmativo se oprime : Y, en caso negativo : N.

```
$ DEL/CON                         (R)
$ File: NOMBRE.EXT;*
```

El sistema manda los mensajes :

```
NOMBRE.EXT; n delete Y or N? (Y ó N)      (R)
NOMBRE.EXT; n-1 delete Y or N? (Y ó N)    (R)
etc.
```

También puede usarse :

```
$ DEL/CON                         (R)
$ File : NOMBRE.*;*               (R)
```

Ahora el sistema recorrerá todas las versiones de cada una de las extensiones del archivo que se indica.
De la misma manera se pedirá :

```
$ DEL/CON                         (R)
$ File : *.EXT;*                  (R)
```

para que el sistema pregunte si debe borrarse alguna versión de los posibles archivos, que tienen la extensión pedida.

Por último con el comando

```
$ DEL/CON *.*;*                   (R)
```

el sistema recorrerá todos los archivos del directorio.

Para mantener la última versión de cada uno de los archivos se usa el comando

```
$ PURGE                          (R)
$ File : NOMBRE.EXT              (R)
```

o bien

```
$ PURGE NOMBRE.EXT               (R)
```

Si se desea pueden mantenerse más de una versión, para lo cual se usará una variante de este comando

```
$ PURGE/KEEP = n                 (R)
$ File : NOMBRE.EXT              (R)
```

El sistema borrará todas las versiones del archivo, salvo las n últimas.

Con el comando

```
$ PURGE *.*                      (R)
```

quedarán las últimas versiones de cada uno de los archivos.

Si se desea que el sistema informe de los archivos que borró, se usará el comando

```
$ PURGE/LOG                      (R)
```

La acción ejecutada por este comando es la misma del comando PURGE, pero el sistema da un mensaje indicando lo borrado.

## VIII.6 COPIA DE ARCHIVOS

El comando "COPY" permite transferir la información de otros directorios a nuestra cuenta, de un archivo a otro y también del directorio a algún subdirectorio. La forma de entrar este comando es :

```
$ COPY                                  (R)
```

El sistema responde :

```
$ From :  [ INMABB] PROGRAMA.BAS        (R)
$ To :    [ INMABB] PROG.BAS            (R)
```

El archivo PROGRAMA.BAS será copiado en el archivo PROG.BAS

```
$ COPY                                  (R)
$ From :  [ INMABB] PROGRAMA.BAS        (R)
$ To :    [ INMABB.SBD ] PROG.BAS       (R)
```

El archivo PROGRAMA.BAS será copiado en el archivo PROG.BAS del subdirectorio SBD.DIR.

Cuando el archivo se encuentre en otro directorio, se procederá de la siguiente manera

```
$ COPY                                  (R)
$ From :  [AREA] F.EXE                  (R)
$ To :    [ INMABB]  F.EXE              (R)
```

En este caso se produce la transferencia del programa F.EXE de la cuenta "AREA" a la cuenta "INMABB".

Si la cuenta "AREA" se encontrara en un disco que no sea aquel que el sistema asume por default ( DRAØ en nuestro caso) debe indicarse. Si "AREA" se encuentra en el disco DRA1 cuando el sistema pregunta "From" debe indicarse este hecho de la siguiente manera:

```
$ From : _DRA1 : [AREA] F.EXE           (R)
$ To :   _DRAØ : [ INMABB] F.EXE        (R)
```

## VIII.7 CAMBIO DE NOMBRE DE ARCHIVOS

El comando "RENAME" cambia la identificación de un archivo, para esto se debe indicar al sistema el nombre actual y extensión del mismo y también el nuevo nombre, de la siguiente manera:

```
$ RENAME                                    (R)
$ From : NOMBRE.EXT                         (R)
$ To : NUEVO NOMBRE.EXT                     (R)
```

Observamos que no se indica el número de versión del archivo, en consecuencia el sistema solamente le cambiará el nombre a la última versión del archivo en cuestión, cuya extensión se indica. Usando "comodines" se puede cambiar el nombre a todas las extensiones y versiones del archivo, en este caso pondremos

```
$ RENAME                                    (R)
$ From :  NOMBRE.EXT;*
$ To :  NUEVO NOMBRE.EXT;*
```

O bien

```
$ From : NOMBRE.*;*
$ To : NUEVO NOMBRE.*;*
```

En el primer caso el sistema cambiará el nombre de todas las versiones de la extensión pedida, en el segundo caso cambiará el nombre de todas las versiones de las posibles extensiones del archivo en cuestión.

## VIII.8 IMPRESION DE ARCHIVOS

El comando "PRINT" permite obtener una copia en papel de un archivo, con el mismo, se transfiere la información que sale por pantalla al usar el comando "TYPE" a la impresora del sistema. Este dispositivo tiene por nombre lógico: LPAØ que el sistema sume por defecto.

Para el uso de la impresora el sistema crea una "cola de espera" identificando los archivos con un número, si se desea saber si hay usuarios en dicha cola se usa el comando

```
$ SHOW QUEUE                    (R)
```

El sistema responde con un mensaje

```
_QUEUE: LPAØ
```

donde indica cual es el último trabajo de la cola (Job n);o bien

```
* Device queue "LPAØ"  Forms = 0, Genprt Lower
```

en caso de estar libre.

Si está ocupado por un usuario imprime el siguiente mensaje

```
* Current Job n NOMBRE DE LA CUENTA NOMBRE,
  Pri = K   Fecha y Hora.
```

El comando "PRINT" se introduce de la siguiente manera:

```
$ PRINT NOMBRE.EXT              (R)
```

La respuesta del sistema es :

```
Job m entered on queue
```

El número m identificará nuestro archivo en la cola de espera. También pueden imprimirse más de una copia de un archivo, usando una variante del comando PRINT.

```
$ PRINT / COPIES = n            (R)
$ File : NOMBRE.EXT             (R)
```

con lo cual obtendremos n copias del archivo pedido.
Una vez que el archivo o los archivos entran en la cola de espera pueden ser sacados de la misma, como conocemos el número de job que se les ha asignado al accionar el comando "PRINT" pondremos

$ DELETE / ENTRY = [n] / LPAØ

para eliminar de la cola el Job n; o bien

$ DELETE / ENTRY = [n,m,...] / LPAØ

si en la cola estuvieran los trabajos n,m,...

## IX NOMBRES LOGICOS

Un programa en el cual es necesario que el sistema lea y/o escriba datos puede ser diseñado de manera que la información sea transferida al mismo de subrutinas o archivos externos cada vez que se ejecute. Esto se conoce con el nombre de independencia de archivos y dispositivos.
El sistema operativo del VAX/VMS permite esta independencia mediante el uso de nombres lógicos, que son una serie de caracteres que se le asigna a cada archivo o dispositivo y que el sistema reconoce cuando se los invoca.
Así el nombre lógico : del disco DRAØ es SYS$DISK
de la biblioteca del sistema es:
SYS$LIBRARY
de la terminal es TT

Mediante el comando $ SHOW LOGICAL (R)
el sistema responde con el siguiente listado.

```
Contents of process logical name table :

SYS$INPUT = _TTB6:
SYS$OUTPUT = LPAO:
SYS$OUTPUT = _TTB6:
SYS$ERROR = _TTB6:
TT = _TTB6:
SYS$DISK = DRAO:
SYS$COMMAND = _TTB6:
SYS$LOGIN = DRAO: [ INMABB]

Contents of group logical name table :
group logical name table is empty
Contents of system logical name table :

SYS$DISK = DRAO:
```

```
SYS$SYSDISK = DRAO:
SYS$SYSTEM = DRAO: [SYSEXE]
SYS$SHARE = DRAO: [SYSLIB]
DBG$INPUT = SYS$INPUT:
DBG$OUTPUT = SYS$OUTPUT:
COB$INPUT = SYS$INPUT:
COB$OUTPUT = SYS$OUTPUT:
COB$CONSOLE = SYS$ERROR:
COB$CARDREADER = SYS$INPUT:
COB$PAPERTAPEREADER = SYS$INPUT:
COB$LINEPRINTER = SYS$OUTPUT:
COB$PAPERTAPEPUNCH = SYS$OUTPUT:
PAS$INPUT = SYS$INPUT:
PAS$OUTPUT = SYS$OUTPUT:
LB = DRAO:
LBO = DRAO:
WK = DRAO:
WKO = DRAO:
SP = DRAO:
SPO = DRAO:
SYS$LIBRARY = DRAO: [SYSLIB]
SYS$MESSAGE = DRAO: [SYSMSG]
SYS$HELP = DRAO: [SYSHLP]
DISK$CONSOLE = _CSA1:
EDT$CAI = SYS$LIBRARY:
SYS$LP_LINES = 68
SYS$PRINT = LPAO
```

Observamos en la lista anterior que algunos nombres lógicos son precedidos por un guión: este corresponde al caracter superior de la tecla correspondiente, por lo tanto debe accionarse previamente "SHIFT".

Cuando se codifica un programa se puede hacer referencia a un archivo de entrada/salida de acuerdo a la sintaxis del lenguaje que se está utilizando.

Una vez que el programa fue compilado, para obtener el programa ejecutable debe hacerse mención de las subrutinas externas (si se hubiera usado alguna) así como del lugar donde se encuentran.

Supongamos que en un programa en lenguaje Fortran DATOS.FOR hubiera una sentencia: EXTERNAL F y que F sea una subrutina que se encuentra en la biblioteca del sistema. Compilamos el programa con el comando:

- FOR DATOS (R)

Encadenamos el programa con el comando:

- LINK DATOS,F, [USER.INSL] SPINSL/LIBRARY (R)

Observamos que en este comando en primer término se da el nombre del programa, en segundo término el nombre del programa de la sentencia EXTERNAL y al final el lugar donde se encuentra y sus características.

Podemos ahora correr el programa DATOS.FOR con el comando

- RUN DATOS (R)

con el cual obtendremos un listado de los resultados por la pantalla de la terminal.

En el programa hay una sentencia del tipo Write (6,n)...

Para el sistema el nombre lógico 6 (del lenguaje Fortran) equivale a SYS$OUTPUT que es el nombre lógico de la terminal de pantalla. Si se desea correr el programa en la terminal impresora para obtener una copia en papel de dicho listado debe asignarse el nombre lógico 6 del programa o sea el nombre lógico SYS$OUTPUT del sistema a la impresora que el sistema reconoce como _LPAØ:, para lo cual deberemos usar el comando "ASSING"

(antes del comando RUN) de la siguiente manera :

```
ASS_LPAØ: SYS$OUTPUT                    (R)
```

O bien

```
ASS                                     (R)
Device: _LPAØ:                          (R)
Logical: SYS$OUTPUT.                    (R)
```

Si ahora usamos el comando

```
-RUN DATOS                              (R)
```

el sistema transfiere el listado que antes apareció por pantalla a la impresora. Pero ahora cualquier comando que se dé será ejecutado por impresora. Para volver a nuestra terminal deberemos hacer la asignación correspondiente.

```
ASS_TTB6: SYS$OUTPUT                    (R)
```

O bien:

```
ASS                                     (R)
Device: _TTB6:                          (R)
Logical: SYS$OUTPUT                     (R)
```

También podemos utilizar el comando

```
$ DEASSING _LPAØ: SYS$OUTPUT            (R)
```

Nuevamente notamos que el nombre lógico está precedido por un guión correspondiente al caracter superior de la tecla correspondiente, por lo tanto al introducir este caracter debe presionarse "SHIFT".

Veamos otro ejemplo: Suponiendo que se tengan archivos de nombres lógicos entrada,salida, que se usan en un programa de nombre PRUEBA, podemos hacer la siguiente asignación

```
$ ASSIGN  DATOS 1.DAT ENTRADA           (R)
$ ASSIGN  DATOS 2.DAT SALIDA            (R)
$ RUN PRUEBA                            (R)
```

El archivo PRUEBA contendrá instrucciones de entrada/salida para abrir leer y escribir usando los archivos de nombre lógico entrada y salida, a saber

```
                .
                .
                .
OPEN 'ENTRADA' , 'SALIDA'
                .
                .
                .
WRITE SALIDA
READ ENTRADA
CLOSE 'ENTRADA' , 'SALIDA'
                .
                .
                .
```

Como vemos las instrucciones del programa prueba son de apertura, escritura y lectura (open,write,read) de archivos en forma general. Al ejecutarlo el sistema lee del archivo entrada y escribe en el archivo salida.
Con la asignación previamente hecha, el sistema lee del archivo DATOS 1.DAT y escribe en el archivo DATOS 2.DAT.

NOMBRES LOGICOS EN LOS COMANDOS

El uso de nombres lógicos no está restringido a la codificación de programas. Algunos comandos como "TYPE" y "COPY" acertan nombres lógicos.
Con la asignación:

```
$ ASSIGN [ INMABB] PERSONAL.LST LISTADO          (R)
```

tenemos la equivalencia entre el nombre lógico "LISTADO" y el archivo "PERSONAL.LST" de la cuenta INMABB. Si ahora usamos el comando

```
$ TYPE LISTADO                                   (R)
```

el sistema muestra por pantalla el archivo "PERSONAL.LST"

NOMBRES QUE EL SISTEMA ASUME POR DEFECTO

SYS$INPUT : Es la fuente de donde el sistema lee los comandos y datos de un programa. La asignación por defecto interactiva es la terminal.

SYS$OUTPUT: Es el lugar donde el sistema escribe las respuestas a los comandos y datos de los programas. La asignación interactiva por defecto es la terminal.

SYS$ERROR : Es el dispositivo en el cual el sistema escribe los mensajes de error e información. La asignación interactiva por defecto es la terminal.

SYS$DISK : La asignación por defecto es el disco DRAØ.

Estos nombres lógicos se pueden usar en la codificación de un programa, así por ejemplo podemos pedir que escriba en el dispositivo de nombre lógico SYS$OUTPUT, la salida se hará por terminal.
Pero también se puede asignar un nombre lógico a otro nombre lógico. Supongamos que en el programa del ejemplo anteiror se hace la asignación

```
$ ASSIGN SYS$OUTPUT  SALIDA                (R)
```

Cuando el sistema ejecuta la sentencia WRITE SALIDA, estos datos los muestra por la pantalla de la terminal.

## X SENTENCIAS DE ASIGNACION

Las asignaciones igualan una cadena de caracteres, valores aritméticos o expresiones a nombres simbólicos. Estas asignaciones y los símbolos que utilizan se usan especialmetne en procedimientos.

Así la sentencia :

```
$ TIME : = SHOW TIME                    (R)
```

asigna el nombre simbólico TIME al valor simbólico "SHOW TIME". En consecuencia la palabra TIME podrá usarse como si fuera un comando: el comando "SHOW TIME"

```
$ TIME                                  (R)
11-DIC-1983   17:20:50
```

También se pueden definir símbolos como sinónimos de comandos del sistema.

En el "LOGIN" de la cuenta INMABB hay definidas algunas asignaciones

```
$ SUB1 : = SET DEFAULT [ INMABB.SUB1]
$ SUB2 : = SET DEFAULT [ INMABB.SUB2]
$ INMABB : = SET DEFAULT [ INMABB]
```

Ahora bien podemos definir un sinónimo del comando "DIRECTORY/FULL" dado por la palabra "LIST"

```
$ LIST : = DIRECTORY/FULL               (R)
```

de manera que si se da el comando

```
$ LIST PRUEBA.TXT                       (R)
```

el sistema lo interpreta como si le hubiera dado el comando

```
$ DIRECTORY/FULL PRUEBA.TXT             (R)
```

El sistema responderá dando una descripción de las características del archivo en cuestión, la protección del mismo y la

cantidad de bloques que ocupa.

CONCATENACION DE SIMBOLOS

Se puede asignar un nombre lógico a una concatenación de símbolos o ítems en una línea de comandos. Para indicarle al sistema que debe ejecutar dicha concatenación cuando se da un comando el nombre simbólico debe encerrarse entre apóstrofes (').

Así el nombre simbólico "PQUALS" puede definirse

```
$ PQUALS : = / COPIES = 2 /FORMS = 4/NOBURST     (R)
```

Si se da el comando

```
$ PRINT REPORT.DAT 'PQUALS'                      (R)
```

El sistema actuará como si se hubiera dado el comando :

```
$ PRINT REPORT.DAT/COPIES=2/FORMS=4/NOBURST      (R)
```

Debe tenerse en cuenta que "COPIES=","FORMS=","NOBURST" son cualificadores del comando "PRINT".

También puede ponerse

```
$ PRINT 'PQUAL' REPORT.DAT                       (R)
```

## XI PROCEDIMIENTOS

Son archivos que contienen líneas de instrucciones y en algunos casos datos.

Su extensión es ".COM" , y podrán ser utilizados como comandos o bien para entradas a programas.

Así podemos tener el siguiente procedimiento que llamaremos "PROMEDIO.COM" con los siguientes comandos.

```
$ FOR PROMEDIO
$ LINK PROMEDIO
$ RUN PROMEDIO
```

Observamos que c/una de las líneas de este archivo comienza con el símbolo "$" que es necesario para la sintaxis de los comandos que el sistema ha de procesar.

Para ejecutar un procedimiento se usa el símbolo "@" (arroba) como primer símbolo del nombre. Dando el comando

```
$ @PROMEDIO                                    (R)
```

el sistema buscará el archivo : "PROMEDIO.COM", lee cada una de las líneas y lo ejecuta, para esto debe tener un programa en lenguaje Fortran, llamado "PROMEDIO" para su compilación, encadenamiento y ejecución.

Ahora bien si se desea tener un procedimiento para compilar, encadenar y ejecutar cualquier programa en lenguaje Fortran crearemos el archivo que llamamos: "CREAC.COM" que contiene:

```
$ FOR 'PROGRAMA'
$ LINK 'PROGRAMA'
$ RUN 'PROGRAMA'
```

Observamos que el nombre que figura después de los comandos está escrito entre apóstrofes ('), esto significa que antes de ejecutar el procedimiento debemos definir el valor del

símbolo "PROGRAMA" para lo cual usaremos la siguiente asignación:

```
$ PROGRAMA : = = PROMEDIO                    (R)
```

y posteriormente

```
$ @CREAC                                     (R)
```

Observamos que la asignación : PROGRAMA:== PROMEDIO" contiene los caracteres " : == " Esto significa que la palabra "PROGRAMA" es un símbolo global.

Un asignación del tipo "LIST:=DIRECTORY/FULL" que sólo contiene los caracteres ":=" indica que la palabra "LIST" es un símbolo local.

Al ejecutar el comando "@CREAR" el sistema considera la asignación previa por la cual busca el programa "PROMEDIO", luego lo compila, encadena y ejecuta.

Observamos que este procedimiento podrá servir para ejecutar cualquier programa, siempre que previamente se haga la asignación correspondiente.

El sistema cuenta con símbolos especiales llamados parámetros que pueden usarse al crear un procedimiento, de los mismos se pueden definir hasta 8 en un mismo procedimiento. Así si en el archivo "CREAC.COM" se pone

```
$ FOR 'P1'
$ LINK 'P1'
$ RUN 'P1'
```

con el comando

```
$ @CREAC PROMEDIO                            (R)
```

el sistema asigna el valor "PROMEDIO" al símbolo "P1" que es el primer y único parámetro usado en el procedimiento, sin necesidad de usar la asignación: P1: == PROMEDIO como se hizo anteriormente.

Debemos observar por último que cuando se tiene un programa en el cual se deben entrar datos por pantalla, habrá que especificar el dispositivo de entrada.

En el directorio del INMABB existe un archivo de nombre LOGIN.COM. En él están listados los comandos que el sistema asume por defecto.

Así tendremos:

```
$ INMABB : == SET DEFAULT [ INMABB].
```

## XI.1 PROGRAMACION DE PROCEDIMIENTOS

Los comandos y asignaciones pueden usarse para programar procedimientos de la misma manera que los lenguajes de alto nivel.

## XII REDEFINICION DE COMANDOS

Durante una sesión con la terminal es posible que los programas o archivos usados se hayan compilado, ejecutado y editado varias veces, de esta manera el sistema al final de la sesión tendrá varias versiones de los mismos. Se puede entonces purgar el directorio antes de terminar la sesión y luego usar el comando "LOGOUT = LO" para despedirse. Se puede entonces crear un procedimiento que incluya a estos dos comandos; que llamaremos "LOG.COM" y que contendrá las líneas:

```
$ PURGE *.EXT
$ LOGOUT
```

Si al terminar la sesión usamos el comando

```
$ @LOG                                (R)
```

el sistema borrará todas las versiones, salvo la última, de todos los archivos que tengan la extensión requerida y después cerrará la cuenta, terminando de esta forma con la sesión.

Ahora bien en este caso también se puede hacer una asignación del tipo

```
$ LOG : == @LOG                       (R)
```

De esta manera el símbolo "@LOG" será sustituído por el sistema por el símbolo "LOG" y para terminar la sesión basta con dar el comando

```
$ LOG                                 (R)
```

## XIII EDICION DE ARCHIVOS EN MODO LINEA

La diferencia entre editar o consultar un archivo en "modo pantalla" o en "modo línea" reside en el hecho que el editor asigna un número a cada línea del archivo en el último caso.

Es necesario en consecuencia el uso de comandos específicos para la edición y/o consulta de los archivos. Estos comandos se introducen presionando "RETURN" cada vez que en la pantalla aparece "*".

Los comandos a usar son los siguientes

EXIT, QUIT

INSERT, INSERT n; INSERT END

TYPE n; TYPE n:m (n< m); TYPEWHOLE

RESEQUENCE

SUBSTITUTE

DELETE n, DELETE n THRU m

Los comandos "EXIT" y "QUIT" nos permiten salir del editor y volver al sistema, la diferencia entre ambos reside en el hecho que con el primero el archivo quedará registrado en la memoria del sistema, mientras que con el segundo no ocurre esto.

Si el archivo ya estaba editado, al llamarlo por el editor para su consulta o corrección, al salir con el comando "EXIT" el sistema creará una nueva versión del mismo, lo cual no sucede si se usa el comando "QUIT".

Como con el uso de cualquiera de ellos salimos del editor, la respuesta del sistema será el signo "$", tendremos entonces

```
* EXIT    (R)      o bien      * QUIT    (R)
$ -                              $ -
```

En ambos casos después de $ aparece el cursor -.

Para comenzar la edición de un archivo usamos el comando "INSERT" (Una vez que en pantalla aparece *, el cursor se ubicará a 16 espacios del márgen izquierdo y la escritura se hará desde allí a la derecha. Cada vez que terminamos un renglón o línea del texto ¡debemos oprimir "RETURN"¡

Una vez que terminamos de escribir el texto, para poder ingresar otro comando debemos oprimir "RETURN" y luego "CTRL/Z" la respuesta del editor será: [EOB] (final del archivo) y en la línea siguiente "*". La situación en la pantalla es :

```
* INSERT        (R)

          +-----------+
          | TEXTO DEL |
          | ARCHIVO   |
          +-----------+

                        (R)

                  CTRL/Z

[EOB]
*
```

Para leer el archivo totalmente o algunas de sus líneas usaremos algunas de las variantes del comando "TYPE".

Cuando se edita un archivo en modo línea el editor asigna un número a cada una de las líneas siguiendo en principio el orden natural, de esta manera si así lo deseamos podemos pedir que aparezca en pantalla una determinada línea, varias consecutivas o la totalidad de ellas.

Con el comando:

```
* TYPE N    (R)        aparece en pantalla

    N                   texto de la línea correspondiente
                        al número N

*_
```

Con el comando

```
* TYPE  N:M      (R)      aparece en pantalla

      N
     N+1
      .
      .          texto de las líneas
      .           numeradas desde N
     M-1            a M (inclusive)
      M
* -
```

Con el comando

```
* TYPE WHOLE     (R)      aparece en pantalla

      1
      2
      .          texto completo del archivo
      .          con sus líneas numeradas
      .
      N

[EOB]
* -
```

Como vemos al terminar de ejecutar un comando aparece en pantalla el "*" que nos indica que el editor está a la espera de un comando.

Podemos ahora agregar líneas al texto, usando el comando "INSERT" o alguna de sus variantes según el lugar donde se desee que estas queden incluídas. La respuesta a cualquiera de estos comandos es que el cursor se ubicará a 16 espacios del márgen izquierdo permitiéndonos ingresar nuevas líneas al archivo.

La diferencia que existe entre estos comandos es la ubicación que el editor dará a las líneas ingresadas. Recordamos que el comando se termina con "RETURN" y "CTRL/Z"!

Con el comando

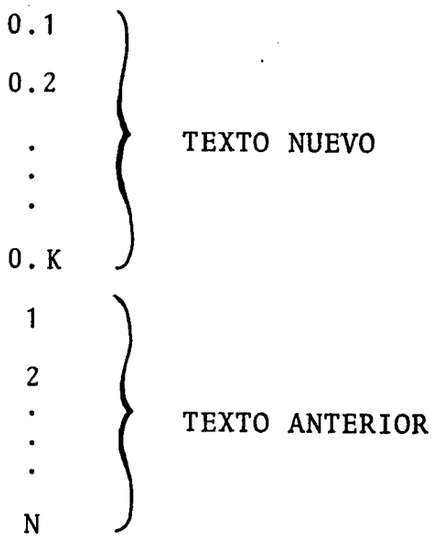
* INSERT (R)

- TEXTO NUEVO

CTRL/Z (R)

al pedir

* TYPE WHOLE (R) en el visor aparecerá




[EOB]

*

Con el comando

* INSERT K (R)

- TEXTO NUEVO

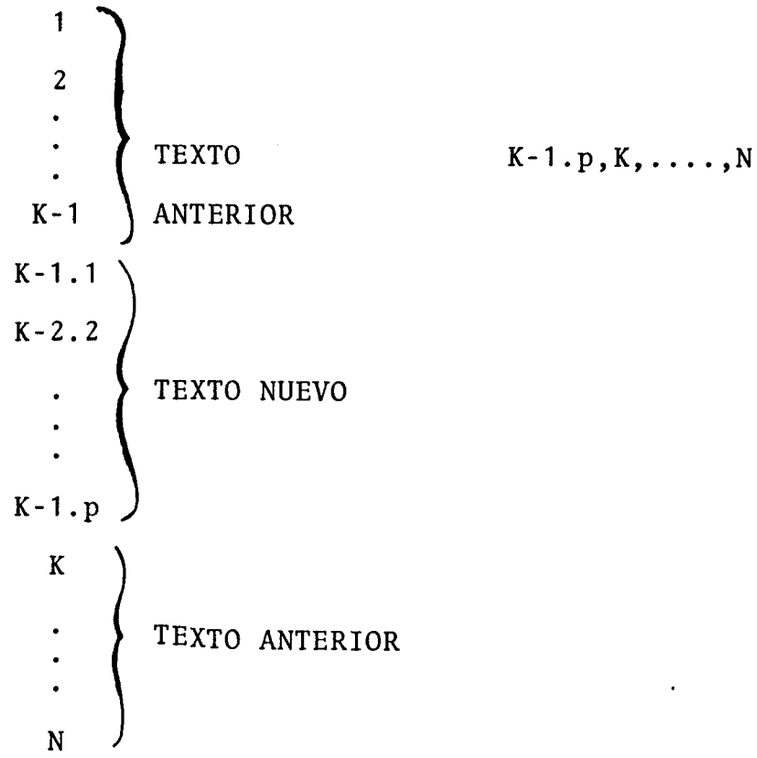
..... (R)

CTRL/Z

K TEXTO DE LA LINEA K (aparecerá en el visor al terminar de escribir el nuevo texto)

Si pedimos

* TYPE WHOLE (R) El sistema muestra en la pantalla el texto del archivo con la siguiente numeración de línea: 1,...,K-1,K-1.1,K-1.2,




[EOB]

*

Observamos que el hecho de haber insertado líneas en el archivo al principio del mismo o entre dos, queda evidenciado por la numeración asignada por el sistema a las mismas: 0.1,0.2,0.3,.... (Antes de la línea 1) en el primer caso, K-1.1,K-1.2,...,K-1.p (Antes de la línea K) en el segundo.

Si queremos que las líneas del archivo sean renumeradas, desde 1 siguiendo el orden numérico usamos el comando

```
* RESEQUENCE        (R)        el sistema imprime el
                                          mensaje
  M lines resequenced
*
```

En este mensaje indica la cantidad de líneas renumeradas.

Para ingresar el nuevo texto a continuación de lo que ya tenemos en el archivo usamos el comando

```
* INSERT END          (R)
              - TEXTO NUEVO      (R)
                . . . . . .
              CTRL/Z
[EOB]
*
```

En este caso a las nuevas líneas el sistema le asignará la numeración que corresponde respecto de la última que tenía registrada. Si pedimos

```
* TYPE WHOLE          (R)        El sistema muestra en
                                    la pantalla
  1  )
  2  )
  .  )  TEXTO ANTERIOR
  .  )
  .  )
  N  )
 N+1 )
  .  )
  .  )  TEXTO NUEVO
  .  )
  M  )

[EOB]
*
```

Con el comando "DELETE" se pueden borrar líneas del archivo. Usaremos

"DELETE N" para borrar la línea N

"DELETE N THRU M" (N < M) para borrar desde la línea N hasta la M inclusive.

La respuesta del sistema a estos comandos son mensajes que indican la acción ejecutada, además muestra la línea que siguen a las que se eliminaron

```
* DELETE N          (R)          Aparece en pantalla

  1 line deleted
      N+1     TEXTO DE LA LINEA N+1
*
```

Con el comando

* DELETE N THRU M (R) El sistema responde:

K lines deleted

M+1 TEXTO DE LA LINEA M+1

Hay que notar que el uso de esta comando hace que el editor elimine del archivo algunas líneas, por lo tanto la numeración de las que quedan presentará los saltos que correspondan.

Cuando estamos escribiendo el texto en el programa o archivo podemos cometer errores de escritura, los mismos pueden corregirse en el momento en que el error fue cometido. El teclado de la terminal tiene varios comandos que permiten efectuar esa acción; a saber

"DELETE" ubicada a la derecha y arriba del comando "RETURN". La acción de este comando permite borrar 1 a 1 las letras que se encuentren a la izquierda del cursor.

Oprimiendo simultáneamente los comandos "REPEAT" y "DELETE" la acción del último será repetida tantas veces como se desea permitiendo borrar más de una palabra o varias letras rápidamente.

Si se desea borrar la línea que se está escribiendo en su totalidad usaremos el comando CTRL/U. Con el mismo el cursor quedará ubicado al principio de línea.

Notamos por último que en caso de ser necesario es posible hacer un reemplazo de palabras en una determinada línea. Pedimos para esto al editor que imprima la línea en cuestión con el comando

* TYPE N (R) En la pantalla aparecerá

N TEXTO DE LA LINEA PEDIDA

Para cambiar una palabra usaremos el comando "SUBSTITUTE" indicando qué palabra se quiere eliminar del texto y la palabra que la sustituye, de la siguiente manera:

* SUBSTITUTE¡ PALABRA A CORREGIR¡ PALABRA NUEVA¡ (R)

La respuesta del sistema será:

```
N   NUEVO TEXTO
1 substitution
*
```

NOTA FINAL:

Los archivos editados en"modo línea" pueden ser consultados en "modo pantalla". La corrección de los mismos podrá hacerse en "modo keypad" o bien en"modo nokeypad".