10
20

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ARAGON

OPTIMIZACION DE TIEMPO Y CODIGO PARA EL DESARROLLO DE SISTEMAS.

T E S I S

Que para obtener el Título de

INGENIERO EN COMPUTACION

p r e s e n t a:

ARTURO CROTTE FRANCO

DIRECTOR DE TESIS

ING. ROBERTO BLANCO BAUTISTA

MEXICO, D. F. NOVIEMBRE 1994

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ARAGON
UNIDAD ACADEMICA

ING. SILVIA VEGA MUYTOY
JEFE DE CARRERA DE INGENIERIA
EN COMPUTACION
P R E S E N T E .

En atención a su solicitud de fecha 24 de octubre del año en curso, por la que se comunica que el alumno ARTURO CROTTE FRANCO, de la carrera de INGENIERO EN COMPUTACION, ha concluido su trabajo de investigación intitulado "OPTIMIZACION DE TIEMPO Y CODIGO PARA DESARROLLO DE SISTEMAS", y como el mismo ha sido revisado y aprobado por usted se autoriza su impresión, así como la iniciación de los trámites correspondientes para la celebración del Examen Profesional.

Sin otro particular, le reitero las seguridades de mi atenta consideración.

ATENTAMENTE
"POR MI RAZA HABLARA EL ESPIRITU"
San Juan de Aragón, Edo. Méx., Octubre 24, 1994.
EL JEFE DE LA UNIDAD

LIC. ALBERTO IBARRA ROSAS

c c p Ing. Roberto Blanco Bautista, Asesor de Tesis.
c c p Interesado.

AIR'eom.

DEDICATORIA :

A MEXICO, PARA QUE ENTRE TODOS HAGAMOS DE EL, UN MEJOR LUGAR.

AGRADECIMIENTO EN ORDEN ALFABETICO :

A TI, POR SI NO ESTAS EN LA LISTA (QUE ESTE TRABAJO TE SEA ÚTIL).

Alejandro Sánchez : Por tu amistad y confianza.

Alex : Por tu amistad y tu lealtad.

Carmen y Olivia : por su amistad y apoyo en los cimientos de mi carrera.

David : por el *SANGRÍA*, el *CREAMOVE*, y tu amistad.

Diohema : por tu apoyo, confianza, y consejos. Pero más por tu amistad.

Don Horacio : Papá ejemplo de firmeza en los objetivos. Descansa en paz.

Don Rolando : gracias suegro por la confianza.

Doña Licha : Mamá gracias por ayudarme a llegar (Y aún nos falta).

Doña Salome : por un millon de detalles, y su confianza, "el té de limón".

Enrique : gracias hermano mayor por las muelas y usar uno de mis sistemas.

Estela Ruiz : gracias por tu amistad y el esfuerzo en serigrafía.

Graciela : gracias hermana por tu ejemplo de decisión y libertad.

Hilda : gracias hermana por tus revisiones a este trabajo.

Homero : por ser un buen amigo (Canito hijo mio).

Horacio : gracias carnal por tu apoyo y consejos.

Imelda : por tu amistad, las lunadas y la confianza.

Ing. David G. Maxines : por ser un buen maestro y un amigo de Icom-84.

Ing. Ernesto : por su paciencia y la visión de darnos una alternativa.

Ing. Roberto Blanco : por sus enseñanzas y la forma en como nos apoya.

Ing. Villagrana: por ser un "INGENIERO" ejemplo que muchos queremos alcanzar.

Isaac Sánchez : por darme la oportunidad, la confianza y tu amistad.

Jorge : gracias hermano por las ayudas automovilisticas (sin ti no llegamos).

Juan Ramon : por tu confianza en mis expectativas. (ojala se cumplan)

Lalo y Luz : por su amistad y confianza.

Lety : mi esposa. Por todo tu amor y por ser tan compatible (mitad - mitad - **pareja**).

Lic. Ayala : por ayudarme a mejorar y no dejarme hacer cosas mediocres.

Lilia : por ser una hermana que ayuda con mano firme.

Martin Esquivel : por las compilaciones parciales, el cuadrito y otras rutinas.

Mary : por la oportunidad de convivir con un ángel. Descansa en paz.

Miguel Angel : (mijo - el gato), por tu amistad.

Minerva : gracias hermana por tus consejos y la plancha.

Nacho : por la confianza.

Noe y Estela : por su amistad, la olímpica acapulqueña y la preluna.

Paco : ! hey Paco !. Por tu ayuda, confianza y amistad, gracias.

Ramon : por tu nobleza, ejemplo que deberíamos seguir todos.

Ricardo Vargas : por la presión y por tu amistad.

Roberto : por tu amistad, la justificación de memos y otras rutinas.

Rolando : por tu amistad y confianza, gracias cuñado.

Sandra : por tu ayuda en los momentos complejos de estudio.

Sra. Carmelita : por su confianza, sencillez, nobleza, apoyo, amistad y mil etcéteras.

Sra. Lupita Santos : por su confianza y amistad, y por ser como otra madre.

Sra. Margarita : gracias... ! a DIOS !, por su confianza, por su familia y mi esposa

Willis : por tu amistad, los remaches, y el sandwich de chile habanero con jamón.

Yeye : por que fuiste una figura para no rajarme en un momento de flaqueza.

*** Í N D I C E ***

INTRODUCCIÓN

OBJETIVOS

Durante el desarrollo del presente trabajo destacan varios objetivos :

- Proveer al lector de los conocimientos fundamentales en el manejo de bases de datos.
- Conocer las características y uso del software dedicado al xBase.
- Distinguir entre lenguajes y herramientas CASE, así como ambientes de trabajo para xBASE, sus intérpretes y compiladores.
- Aprender lo relacionado al análisis y diseño de sistemas, así como los métodos que facilitan el desarrollo y buen desempeño de un sistema.
- Introducir al lector en el uso del lenguaje xBase y sus características.
- Asistir al lector en las tareas básicas de manejo de datos en xBase

Y por último y PRINCIPAL objetivo.

- Conjuntar los conocimientos de : lenguaje xBase, Herramientas CASE, análisis y diseño de sistemas y los Amabientes xBase, para crear un solo programa xBase con características hereditarias y facilidad de adaptación para el manejo óptimo y natural de cualquier base de datos.

JUSTIFICACIÓN

La importancia de realizar esta propuesta de tésis enfocada a los sistemas xBase, radica principalmente en :

-La cantidad de usuarios que actualmente recurren a las bases de datos.

-Al tiempo que les toma consultar y capacitarse dentro de las mismas, por la carencia de una interfase natural.

-Y sobre todo, al tiempo y costo que se invierte en desarrollo de cada nueva tarea.

Aunque el enfoque principal de esta tésis va dirigido a CLIPPER, es importante mécionar que la teoría de movimiento (MOVE) es aplicable a cualquier producto manejador de bases de datos, con programación xBase.

Se enfoca a Clipper por ser un compilador y precompilador bastante eficiente que permite generar los programas ejecutables en un tamaño promedio más pequeño que otros productos, al mismo tiempo que, EN TIEMPO DE EJECUCIÓN los sistema bajo clipper son relativamente más rápidos.

HIPÓTESIS

Uno de los principales temas durante nuestra educación como INGENIEROS UNIVERSITARIOS, ha sido la OPTIMIZACIÓN, que es el factor principal que define a un ingeniero, además de la inventiva y los conocimientos científicos.

El planteamiento central de esta tésis es precisamente OPTIMIZAR, ya que durante mucho tiempo hemos desarrollado sistemas con los algoritmos básicos, sin ir más alla y buscar un módelo que nos permita heredar la mayor cantidad de código a las aplicaciones futuras, ahorrando con esto un largo tiempo, que es posible dedicar a mejores investigaciones y mayor optimización.

DESCRIPCIÓN GENERAL

El CAPÍTULO I:

Esta enfocado a cimentar las bases de conocimiento para sistemas xBase, dotar al lector de una herramienta sencilla (no optimizada) para generar sistemas con dBase IV, y es así mismo, una guía de auto enseñanza.

La segunda parte proporciona una introducción al lenguaje xBase, al mismo tiempo que OPTIMIZA el código generado en la primera parte.

El CAPÍTULO II:

Abarca desde las estructuras básicas de programación en lenguaje xBase, los diferentes ambientes y compiladores, ventajas y desventajas de cada uno, hasta los métodos avanzados de compilación dinámica.

El CAPÍTULO III:

Plantea las bases de un buen resultado basado en los conceptos fundamentales del Análisis y Diseño de Sistemas, así como la mejor utilización de los diagramas de Flujo, Bloques y Warnier-Jackson dentro del desarrollo de sistemas.

El CAPÍTULO IV:

Es el planteamiento central de este trabajo. Consiste en hacer una recapitulación óptima de Algoritmos, Métodos, Subrutinas, Procedimientos, Funciones, Diagramas, Tips y Experiencias para desembocar en la teoría de movimiento (MOVE).

El CAPÍTULO V :

Por un lado plantea los objetivos y da la explicación del sistema inicialmente dirigido a PRESUPUESTOS y que por si mismo, dio pie a : Pedidos, Remisiones, Facturación y Ordenes de Compra abarcando así el proceso administrativo global de varias empresas. Y por otro, ejemplifica las ventajas de utilizar los (MOVE) en uno de los sistemas que están ya en operación basados en esta teoría.

El APÉNDICE A:

Esta formado por toda la librería de Programas, Procedimientos, Subrutinas y Funciones que componen el presente trabajo, identificados cada uno por su título.

EL APÉNDICE B:

Conjunto de programas principales que conforman el sistema de PRESUPUESTOS.

El GLOSARIO :

Comprende los términos, abreviaturas y conceptos empleados en esta obra y algúnos otros conceptos.

CAPÍTULO I

MANEJADORES DE BASES DE DATOS

TEMAS DEL CAPÍTULO

1.-ANTECEDENTES.

2.- INTRODUCCIÓN.

3.- MANEJADORES DE BASES DE DATOS.

4.- DBASE IV.

5.- VENTAJAS, DESVENTAJAS Y UNA ALTERNATIVA.

6.- ESTRUCTURAS DE PROGRAMACIÓN XBASE.

7.- CONCLUSIONES DEL CAPÍTULO .

I.1 ANTECEDENTES

Un sistema de información es el conjuto de programas y archivos de datos relacionados entre sí, para dar solución a un problema o lograr algún objetivo, en general los sistemas de información son una herramienta en la toma de decisiones.

Los sistemas de información enfocados al manejo de bases de datos, son creados utilizando un lenguaje de programación para la escritura y construcción de varios procedimientos. Estos programas generalmente son integrados por una serie de algorítmos, los cuales permiten a los usuarios realizar los diferentes procesos para los que fue creado el sistema.

Los programas se pueden representar a grandes rasgos en tres pasos:

Entrada	-	Proceso	-	Salida

Para la selección de opciones dentro del sistema de información se utilizan programas conocidos como menús, en los que se le permite al usuario seleccionar de entre varias opciones la que en ese momento desea ejecutar. Ejemplo:

Selección de opciones usando un Menú alfabetizado o numerado.

```
Menú Principal

A) Materiales
B) Herramientas
C) Maquinaria
D) Terminar

Seleccione una Opción_
```

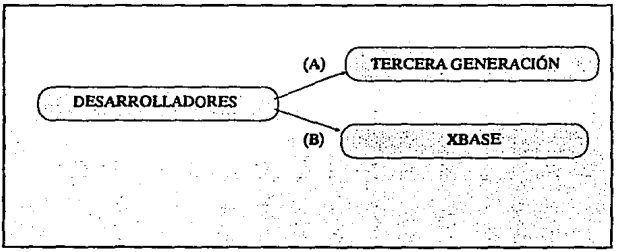
```
Menú de Materiales

A) Altas
B) Bajas
C) Cambios
D) Consultas
E) Reportes
F) Salir

Seleccione una Opción_
```

En este tipo de sistemas el usuario sólo tiene que pulsar una tecla para relizar la acción deseada.

I.2 INTRODUCCIÓN:

En los últimos años se ha dado un giro muy importante en cuanto a los sistemas de información basados en computadoras. Por un lado, la enorme demanda de sistemas manejadores de grandes volúmenes de datos, y por otro, las facilidades que estos deben presentar no sólo a los usuarios finales sino también, a los desarrolladores de sistemas.

Para dar solución a estos problemas podemos tomar la primer bifurcación en el camino :



A) Desarrollar en lenguajes tradicionales de tercera generación o usar lenguajes de bajo nivel.

- BASIC.
- PASCAL.
- COBOL.
- FORTRAN.
- C.
- ENSAMBLADOR.
- ETC.

B) Utilizar manejadores de bases de datos con un lenguaje de programación.

- DBASE II, DBASE III, DBASE III + , DBASE IV (1.0, 1.1, 1.5, 2.0).
- FOX BASE, FOXBASE +, FOX PRO (1.0, 1.2, 2.0, FOR WINDOWS).
- FOX BASE FOR UNIX.
- BTRIVE.
- PARADOX.
- ORACLE.
- ETC.

En esta primer bifurcación tomaremos el camino (B) que por cierto es el camino que la mayoría de los desarrolladores se han decidido a tomar, por la facilidad y seguridad que estos presentan al manejar grandes volúmenes de información.

Es importante recordar que cada lenguaje tiene su ambiente y área determinada y no debemos pensar que por aquí seleccionar la opción (B) estamos menospreciando el trabajo de los otros lenguajes. De hecho en el desarrollo de la presente investigación son utilizados dos de estos lenguajes para dar complemento y fuerza al tema aquí expuesto agradeciendo así a las aportaciones del ING. DAVID HERNANDEZ GOMEZ y comprobado la importancia de los lenguajes de tercera generación como BASIC y PASCAL. De los que más adelante explicaremos su intervención en este trabajo.

I.3 MANEJADORES DE BASES DE DATOS

CONCEPTOS FUNDAMENTALES.

MANEJADORES DE BASES DE DATOS : Son los sistemas desarrollados con el objetivo de crear controlar y manipular bases de datos.

Base de datos : Archivo aleatorio con estructura de campos y registros.

Sistema : Programa o conjunto de programas con un fin determinado y relacionados entre sí para lograr un fin u objetivo determinado.

Programa : Secuencia de INSTRUCCIONES escrita en algún lenguaje de computadora.

Archivo : Almacén de información.

Archivo Aleatorio : Aquel que permite lectura y escritura directamente sobre un registro cualquiera.

Información : Conjunto de datos con un orden lógico.

Dato : Unidad de información.

Campo : Espacio o conjunto de datos de una misma especie.

Registro : Conjunto de campos que forman un renglón de información.

Actualmente existen como ya se ha mencionado, un gran número de sistemas llamados comercialmente, PAQUETES, enfocados al manejo de bases de datos. A continuación nombraremos algunos de ellos y sus características principales.

DBASE IV 1.5

FOXPRO/LAN 2.0

ORACLE 7

PARADOX 4.0

DATAEASE 4.2

KNOWLEDGE MAN 3.0

R:BASE 3.5

SUPERBASE 4.0

= = = = = = CARACTERÍSTICAS GENERALES DE LOS PRODUCTOS = = = = = = =

DATAEASE :

640K DE RAM

1.4 MB EN DISCO

S.O. 3.1 O SUPERIOR

NO. DE CAMPOS 255

DBASE IV 1.5 :

640K DE RAM

3.5 MB EN DISCO

S.O. 3.1 O SUPERIOR

NO. DE CAMPOS 255

FOXPRO/LAN 2.0 :

512K DE RAM

14 MB EN DISCO

S.O. 3.1 O SUPERIOR

NO. DE CAMPOS 255

KNOWLEDGE MAN 3.0 :

640K DE RAM

5 MB EN DISCO

S.O. 3.1 O SUPERIOR

NO. DE CAMPOS 255

PARADOX 4.0 :

640K DE RAM

6 MB EN DISCO

S.O. 3,1 O SUPERIOR

NO. DE CAMPOS 255

R:BASE 3.1 :

640K DE RAM

2 MB EN DISCO

S.O. 3.1 O SUPERIOR

NO. DE CAMPOS 255

SUPERBASE 4.0 :

640K DE RAM

2 MB EN DISCO

S.O. 3.1 O SUPERIOR

NO. DE CAMPOS 255

ORACLE 7:

640K DE RAM

16 MB EN DISCO

S.O. 3.1 O SUPERIOR

NO. DE CAMPOS 255

Los anteriores son sólo algunos de los manejadores de base de datos que son utilizados por un elevado número de compañías y usuarios tanto en E.U.A.. como en nuestro país, asimismo existirán otros con características similares o superiores y como ya es sabido por todos nosotros... La carrera continúa.

Tal es así que a la fecha ya están a nuestra disposición, versiones que trabajan bajo ambiente WINDOWS, como son :

PARADOX FOR WINDOWS

FOXPRO FOR WINDOWS

Todos estos manejadores de bases de datos, merecen reconocimiento por las facilidades que nos brindan, en cuanto al control y seguridad en :

La creación de tablas de archivos aleatorios.

La modificación de las mismas.

El control de los apuntadores de principio y fin de archivo.

El apuntador de registro actual y total de registros.

Las facilidades de agregar, insertar o eliminar registros.

La generación de INDICES.

La creación y edición de pantallas de captura.

La creación y edición de reportes a la medida.

La creación y edición de etiquetas (engomados).

Los programas generadores de PROGRAMAS.

Por mencionar algunas de las inumerables ventajas de todos estos PAQUETES, que parece serán interminables.

Tomando como base el VOX-POPULI, aceptaremos que el iniciador de toda esta revolución de información fue basado en el sistema VULCANO desarrollado para el proyecto APOLO 11 de la U.S.A. NASA en el año de 1968, con el objetivo de manejar la información referente a las naves espaciales en cuestión y a cada una de las partes que las componían. De ahí se desprendío un producto llamado DBASE II, de dominio público comercialmente explotado en esa época en las microcomputadoras APPLE y APPLE II/E, posteriormente DBASE III hacia 1981 - 82 en las PC-XT - Jr. de IBM. Más adelante en 1985 se libera DBASE III PLUS, que marca durante los últimos años el estandard en estructuras XBASE, hasta las últimas versiones de DBASE IV, FOXPRO, PARADOX, EASEBASE, SUPERBASE, ETC. Continuando a la fecha en todas las computadoras personales compatibles.

Siendo dBase un producto que tiene a la fecha un enorme número de seguidores y posiblemente sea el más comercial cuando menos en la plataforma de las computadoras personales, veremos a continuación sus características y como aprovechar todo el potencial que este nos ofrece, haciendo referencia a que la mayor parte de los puntos que revisaremos pueden asimismo plicarce en los demás productos manejadores de bases de datos.

De este modo iniciaremos por dar un acercamiento a las características y uso del DBASE IV.

I.4 DBASE IV

I.4.1 CARACTERÍSTICAS Y REQUERIMIENTOS

PC COMPATIBLE, XT, AT o PS/2

640 KB / RAM

3.5 MB DISCO DURO

S.O. D.O.S. 3.1 o posterior

S.O. OS/2 1.0 o posterior

Cualquier configuración de RED que sea 100% compatible con NETBIOS 3.1 o posterior.

NÚMERO MÁXIMO DE REGISTROS	1000 MILLONES
NÚMERO MÁXIMO DE CAMPOS	255
NÚMERO MÁXIMO DE CARACTERES EN UN CAMPO	254
NÚMERO MÁXIMO DE CARACTERES EN UN CAMPO MEMO	64000
NÚMERO MÁXIMO DE DIGITOS NUMERICOS POR CAMPO	20
NÚMERO MÁXIMO DE BASES ABIERTAS SIMULTANEAMENTE	10
NÚMERO MÁXIMO DE ARCHIVOS ABIERTOS	99
DÍGITOS EN EXPRESIONES LOGICAS (.Y.,.T.,.N.,.F.)	1

FORMATOS DE FECHA : MM/DD/YY , DD-MM-YY , YY.MM.DD Y OTROS

I.4.2 INSTALACIÓN :

DBASE IV como cualquier otro paquete necesita de una INSTALACIÓN en disco duro la cual es muy rápida y sencilla, en la que sólo tendremos que correr el programa INSTALAR del disco de instalación de DBASE IV y reemplazar los discos según sea solicitado, además de seleccionar algunas opciones como :

PATH o ruta donde será instalado (C:\DBASE)

Utilizar memoria CACHE (sí - no)

Instalar SAMPLES (sí - no)

Instalar TUTORIAL (sí - no)

Y por último realizara las modificaciones correspondientes a los archivos AUTOEXEC.BAT y CONFIG.SYS, en los que encontraremos los siguientes cambios :

Agregará al PATH del AUTOEXEC.BAT la ruta donde instaló DBASE

PATH %C:\DBASE;%PATH

Modificará en el CONFIG.SYS el número de files y buffers

FILES = 30

BUFFERS = 25

Después de ésto, podemos empezar a trabajar con dbase, pero recomendamos primero ejecutar el programa DBSETUP para seleccionar el tipo de IMPRESORA que utilizaremos así como otras características de personalización del paquete como : COLORES, DISPLAY 25/43, FORMATO ($), SEGUNDA IMPRESORA, etc.

Este programa lo podremos utilizar en cualquier momento para realizar cambios al ambiente de trabajo de DBASE IV.

I.4.3 PRIMERA SESIÓN DE TRABAJO

NOTA :

En la escritura de los comandos o instrucciones siguientes encontraremos :

[ ] para indicar OPCIONAL

{ } para indicar OBLIGATORIO

expC Expresión Caracteres

expN Expresión Numérica

expD Expresión Fecha

expL Expresión Lógica

Para salvar datos o pantallas se usarán conjuntamente las teclas :

Ctrl + W o Ctrl + End.

Antes de iniciar nuestra primera sesión es importante generar un subdirectorio donde almacenaremos todos nuestros archivos.

ej.

CD DBASE

MD DBDATOS

Después de crear el subdirectorio de datos, es recomendable crear un archivo de ejecución por lotes para facilitarnos el acceso diario a nuestra zona de trabajo.

ej. DB4.BAT

```
ECHO OFF
CD\DBASE\DBDATOS
DBASE [/T] [ programa.prg ]
DEL *.$*
DEL TMP*.*
CD\
CLS
```

Ahora que ya contamos con todos lo elementos para poder trabajar es necesario reiniciar (REBOOT) a nuestra computadora para que reconozca los parámetros actuales del CONFIG.SYS Y AUTOEXEC.BAT

Ya que hemos reiniciado podemos ejecutar ahora el DB4 .

Estaremos ahora en el DBASE IV CONTROL CENTER y el informe de catálogo debera ser: CATALOG C:\DBASE\DBDATOS\UNTITLED.CAT esto nos indicará que estamos trabajando en el subdirectorio DBDATOS con un catálogo sin título, el cual podemos modificar más adelante. Por el momento haremos una explicación del CONTROL CENTER sus características y el modo de usarse.

(pantalla del CONTROL CENTER)

Catalog Tools Exit 12:00:00 pm

dBASE IV CONTOL CENTER

CATALOG: C:\DBASE\DATOS\UNTITLED.CAT

Data	Queries	Forms	Reports	Labels	Aplications
<create>	<create>	<create>	<create>	<create>	<create>

File : New file

Description : Press ENTER on <create> to create a new file

Help:F1 Use: Data:F2 Design:Shift-F2 Quick Report:Shift-F9 Menus:F10

En la primer línea de la pantalla podemos ver las opciones del menú (CATALOG TOOLS EXIT) mismas que pueden ser accesadas por la tecla F10 o por medio de la tecla {Alt} + {C}, {T} o {E} iniciales de cada opción. Después aparece la HORA, y el letrero resaltado de esta pantalla "DBASE VI CONTROL CENTER".

En la siguiente línea está el indicador del catálogo el que ahora podemos cambiar utilizando la opción (alt) para elegir USAR OTRO CATÁLOGO y crear uno como EJEMPLO.CAT al salir de esta opción nos pedirá la descripción del catálogo (Documentación).

Ahora pongamos nuestra atención en las 6 divisiones de la pantalla, pues en ellas se almacenarán los archivos que vamos a CREAR o MODIFICAR.

DATA QUERIES FORMS REPORTS LABEL APLICATIONS

Las extensiones directamente asociadas a éstas (no aparecen) son:

Data	Queries	Forms	Reports	Labels	Applications
<create>	<create>	<create>	<create>	<create>	<create>
DBF	QBE	SCR	FRM	LBL	APP
DBT	QBO	FMT	FRG	LBG	PRG
NDX		FRO	FRO	LBO	BAR
MDX					POP
					DBO

Casi al final tenemos las líneas que nos indican el archivo que está en uso y la descripción del mismo.

En la última línea de la pantalla tenemos los Mensajes (que por cierto como usuarios es la línea que NO LEEMOS) y nos indica las teclas de:

Help:F1 Use: Data:F2 Design:Shift-F2 uick Report:Shift-F9 Menus:F10

La navegación dentro de esta pantalla se hará con las teclas de cursor y la tecla Enter para aceptar una opción.

Para crear una base de datos es indispensable conocer los tipos de campos de que podemos disponer. Los nombres de los campos pueden ser de hasta 10 caracteres iniciando siempre con letra, y sólo soportan el bajo guión como caracter especial. Las especificaciones de soporte en cada campo son las mencionadas anteriormente.

I.4.4 TIPOS DE CAMPOS

CAMPO	TIPO
CHARACTER	ALFANUMÉRICO
NUMERIC	NUMÉRICOS
DATE	FECHAS
LOGICAL	LÓGICOS
MEMO	TEXTOS
FLOAT	PUNTO FLOTANTE

I.4.5 CREANDO BASES DE DATOS

Para crear una base de datos solamente tendremos que pulsar ENTER en la opción Create de la Sección DATA . En este momento aparece una tabla que nos permitirá editar las características que queremos definir a nuestra base de datos indicando :

NOMBRE DEL CAMPO

TIPO DE CAMPO

ANCHO (CANTIDAD DE CARACTERES O DÍGITOS CONTANDO DECIMALES Y .)

DECIMALES

INDEXADO (si o no) Manejará un índice Múltiple .MDX

Recomendamos CREAR ahora 2 bases de datos como ejemplos. Una para guardar la información correspondiente a las personas de nuestro lugar de trabajo incluyendo una clave para el puesto que estas ocupan, y otra base de datos que contendrá los posibles puestos en que se pueden desempeñar, así como el importe del sueldo que corresponde a cada puesto, en la que aparezca el campo clave del puesto.

I.4.6 CAMPOS RELACIONALES

El término CAMPO RELACIONAL se refiere a aquellos campos que estarán en diferentes bases de datos y que serán utilizados para hacer referencias cruzadas de una a otra base de datos. Por ejemplo si queremos saber cúales personas pertenecen a un determinado puesto o más aún, si deseamos cambiar el sueldo a un puesto y que como consecuencia de esto a todas las (N) personas asignadas, a éste, sean afectadas de igual forma, y que para este movimiento sólo nos tome el modificar el sueldo al registro del puesto deseado, entonces debemos utilizar CAMPOS RELACIONALES.

Estos deberán cumplir con ciertas características como :

1) Deberán ser del mismo tipo

ej. Character - Character

Numeric - Numeric

2) Deberán tener la misma longitud.

ej. Character 10 - Character 10

Numeric 4 - Numeric 4

Si en los ejemplos propuestos, los campos de clave de puesto no cumplen con estos requisitos, podemos ahora modificar la estructura de dichas bases para poder usar la característica de las BASES DE DATOS RELACIONALES.

I.4.7 CREANDO UN QUERIE

Iniciaremos esta sección explicando el concepto QUERY (pregunta) o como se conoce en la jerga, PETICIÓN.

El uso de un QUERIES va directamente relacionado con la petición de información de una o más bases de datos, seleccionando de ellas los campos que se desea utilizar con la o las condiciones que sean necesarias. A continuación mencionaremos algunos ejemplos de esto.

1.-Se usa un QUERIE para una base de datos cuando, solamente deseamos hacer consulta sin modificación de la información en algunos campos que sean de interés para esta consulta y además, para aquellos registros que cumplan cierta condición.

2.- Se usa un QUERIE para dos o más bases de datos cuando, queremos visualizar la información que le corresponde a cada registro con respecto a las otras bases de datos. Para esto es necesario que exista por lo menos un campo común entre cada base de datos. Asimismo podemos también agregar la condición o condiciones que se requieran.

Para crear un QUERIE solamente se deberá pulsar enter en la opción Create de la sección QUERIES . Si tenemos una base de datos en uso, el esqueleto (SKELETON) de la misma aparece en la parte superior de la pantalla y también en la vista (VIEW) de la parte inferior.

El VIEW es la parte donde podemos quitar y poner con la tecla F5 (TOGGLE), los campos que deseamos de las bases de datos (SKELETONS) de la parte superior.

En caso de que no estuviera la base de datos en uso utilizaremos la opción de menú LAYOUT para adicionar un archivo al QUERIE, seleccionar la base PERSONAS y aparezca el SKELETON en la parte superior pero no así en el VIEW, usando F5 podemos bajar o subir los campos del SKELETON al VIEW.

Ahora repitamos la operación para traer el SKELETON de la base de datos PUESTOS, poniendo en el VIEW con F5 los campos PUESTO y SUELDO.

Para poder hacer la relación entre las bases en uso, podemos hacerlo con la opción CREATE LINK BY POINTING del menú LAYOUT, o simplemente escribiendo una palabra como LIGA abajo de los campos a relacionar en este caso CLAVE DEL PUESTO, creando así dos bases de datos RELACIONALES.

Para agregar a esto una condición usaremos la opción CONDITION y agregaremos una caja de condición (CONDITION BOX) en la que escribiremos la condición.

ej. SUELDO 1200

Es importante saber que usando las teclas SHIFT F1 los usuarios inexpertos pueden construir las condiciones que requieran, dado que esta ayuda maneja : CAMPOS, OPERADORES Y FUNCIONES con las que se construyen las CONDICIONES. Mencionando también que la capacidad de caractéres de una condición es de 1Kb y se puede visualizar mejor con la tecla F9 ZOOM.

Para verificar si el CONDITION BOX es correcto solamente debemos pulsar enter para que DBASE analice si la sintaxis es correcta, indicando en caso de error en la parte inferior de la pantalla.

Por último solamente debemos salvar Ctrl + W, asignar un nombre al QUERIE ej. PERSPUES (PERSonas - PUEStos) y si el QUERIE no contiene errores, se generarán dos archivos :

PERSPUES.QBE - Contiene la programación editable en código DBASE

PERSPUES.QBO - Contiene el programa objeto de ejecución

Ahora podemos hacer uso de la relación PUESPERS pulsando enter en el CONTROL CENTER en el nombre de la misma y seleccionando DISPLAY DATA.

NOTAS :

1.- Si al realizar lo anterior no aparecen datos en la pantalla, posiblemente será porque no existe ningún registro en la bases de datos o los registros no se relacionan con la CLAVE DEL PUESTO.

ej.

PERSONAS		PUESTOS	
nombre	puesto	clave puesto	
MARIA	1	3	SECRETARIA
LETICIA	2	4	JEFE DE PROYECTO

2.- Si en el resultado del QUERIE, a un registro se le asocian dos puestos es porque existen en la base PUESTOS dos registros con una misma clave de puesto.

ej.

PERSONAS		PUESTOS	
nombre	puesto	clave puesto	
MARIA	1	1	SECRETARIA
LETICIA	2	1	JEFE DE PROYECTO

En este caso, podríamos verlo como un error (corregir), pero si se tratara de partidas en una factura lo veríamos como una característica muy aprovechable (INTERPRETACIÓN DE RESULTADOS).

I.4.8 CREANDO FORMS - PANTALLAS DE EDICIÓN Y VISTA

Una de las ventajas más grandes que nos ofrece DBASE IV es la facilidad de crear pantallas de captura, edición o visualización, usando un sencillo método de movimiento de datos y textos. La creación de FORMS se basa en las siguientes utilidades :

- Proveer de un Esquema Rápido de captura (QUICK LAYOUT).

- Enmarcar áreas usando línea DOBLE, SENCILLA o algún CARACTER.

- Con la facilidad de acomodar los TEXTOS y CAMPOS usando :
  - F6 Seleccionar área.
  - F7 Mover área.
  - F8 Copiar área.

- Selección de COLORES para cualquier zona de la pantalla.

- Incluir nuevos campos CALCULADOS.

- Incluir Funciones como : RECN(), RECC(), DBF(), SELE(), etc.

- Modificar formatos numéricos ($9,999,999,999.99).

- Múltiples opciones de editar la captura (PICTURE).

- Opciones de Edición en Cada Campo.

- Permitir captura Si o No.

- Captura con condición IF.

- Mensaje para describir que capturar o que hacer.

- Acarreo del último valor por campo en cada alta.

- Valor DEFOULT en cada campo.

- Valor MÍNIMO.

- Valor MAXIMO.

- Rango de valores ej. EDAD 18,30.

- Generar una condición o PROGRAMA para aceptar datos (WHEN).

- Validación de datos cada vez que se sale del campo

- Mensaje de NO ACEPTADO

- Captura de campos MEMO en VENTANAS o FRAMES

Además de éstas existen algunas otras ventajas que nos facilitan la edición de formatos de captura (FORMS) y que solamente nos tomará el tiempo necesario para formar la presentación final de nuestras pantallas de captura.

Como en la generación del QUERIE solamente tenemos que salvar esta pantalla y asignar un nombre, ej. PERSFORM Para que DBASE IV genere en este momento tres archivos.

PERSFORM.SCR - Contiene todo lo que diseñamos en pantalla.

PERSFORM.FMT - Contiene la programación editable en código DBASE.

PERSFORM.FMO - Contiene el programa objeto de ejecución.

I.4.9 EL GENERADOR DE REPORTES

Es aquí donde DBASE IV nos ofrece una verdadera potencia en el desarrollo de sistemas. En las versiones anteriores el generador era demasiado rígido para considerarlo como una buena herramienta, ya que sólo se tenía un determinado número de líneas para cabeceras y pies de página, además de adolecer de un SCROLL para ubicar el resultado de nuestro trabajo.

Ahora con el nuevo generador no sólo se superaron estas deficiencias sino que además se implementaron TRES cuerpos básicos de reportes : COLUMNAS, FORMATOS y CORREO

Y para cada una de ellas, se integran las siguientes BANDAS de reporte complementándose de la siguiente forma :

CABECERA - HEADER BAND

INTRODUCCION - INTRO REPORT BAND

DETALLE - DETAIL BAND

SUMARIO - SUMMARY REPORT BAND

PIE DE PAGINA - FOOTER BAND

AGRUPAMIENTO - GROUP BAND

Cada una de ellas puede ocupar los renglones que el usuario guste, considerando solamente el número total de renglones definidos en nuestro papel de impresión, así como lo que le restamos a cada una de las otras BANDAS.

CABECERA - Aparecera en todas las páginas del reporte.

INTRODUCCIÓN - Solamente se escribe en la PRIMER página

DETALLE - Esta banda define el CUERPO BÁSICO del reporte.

SUMARIO - Solamente aparecera en la última página.

PIE DE PAGINA - Aparece en todas la páginas del reporte.

AGRUPAMIENTO - Esta banda sirve para provocar un salto de hoja cada vez que ocurre un cambio de datos para el CAMPO especificado en la banda. ej. PUESTO.

Los tres tipos de reporte varian básicamente en la banda de detalle y se distinguen de la siguiente manera.

COLUMNAS - Cada registro ocupa el número de renglones que usemos en esta banda.

FORMATOS - Cada registro ocupa un formato diseñado en pantalla y proporcionalmente el número de formatos para cada hoja.

CORREO - Cada registro ocupará una o más hojas, dependiendo la longitud del texto escrito en esta BANDA, pudiendo mezclar CAMPOS en este.

De ésta forma podemos crear al igual que en FORMS el tipo de salida que necesitamos, además que nos permite el envío de reportes a PANTALLA, ARCHIVO EN DISCO, PUERTOS PARALELOS y PUERTOS SERIE, lo que nos da un potencial multiplicado.

Cuando terminemos de diseñar un reporte y lo salvemos asignándole un nombre PERSREPO serán generados por DBASE tres archivos.

PERSREPO.FRM - Contiene todo lo que diseñamos en pantalla.

PERSREPO.FRG - Contiene la programación editable en código DBASE.

PERSREPO.FRO - Contiene el programa objeto de ejecución.

I.4.10 LAS ETIQUETAS

Esta es una utilidad sencilla de usar pero no simplista que DBASE nos ofrece. Y consiste en generar aquel complejo programa que establece las columnas y registros en un formato de etiquetas (engomados) que frecuentemente es difícil encontrar la misma presentación de engomados para no tener que reescribir el programa de impresión.

Existen predefinidos en DBASE los tamaños más comunes para este fin, y si el que tenemos en nuestras manos no es de un tamaño predefinido, podemos definir el Número de caracteres a lo ancho y a lo largo, así como el número de columnas y la separación entre etiquetas.

De este modo, podemos ver aquí un punto en favor de los desarrolladores, ya que para diseñar una etiqueta sólo se necesitan tres pasos:

1.- Establecer el tamaño, columnas y la separación entre éstas.

2.- Llamar a cada campo dentro del cuadro de la etiqueta.

3.- Salvar y asignar un nombre al archivo. ej. PERSETIQ

Generándose también tres archivos que son :

PERSETIQ.LBL - Contiene todo lo que diseñamos en pantalla.

PERSETIQ.LBR - Contiene la programación editable en código DBASE.

PERSETIQ.LBO - Contiene el programa objeto de ejecución.

Al igual que en los reportes podemos enviar las etiquetas a PANTALLA, ARCHIVO EN DISCO, PUERTOS PARALELOS y PUERTOS SERIE, además de que nos permite la impresión de un ejemplo de etiqueta para ajustar y cuadrar que todas queden dentro de los engomados.

I.4.11 EL GENERADOR DE APLICACIONES

LENGUAJE DE 4a GENERACIÓN

Una APLICACIÓN, es el conjunto de programas con fin determinado.

Una aplicación es también, un conjunto de programas que permitirá a los usuarios finales manejar la información de las bases de datos con sus correspondientes archivos, como formas de captura, reportes, etiquetas, relaciones y manejo de índices, sin que éste tenga conocimiento del cómo fueron creadas.

Otra característica es que para realizar un sistema usando el GENERADOR DE APLICACIONES, nosotros tampoco tenemos que tener conocimiento del lenguaje de programación. Fundamento de las herramientas CASE (Computer Assist System Engineering) donde se ubican los lenguajes de 4a. generación.

En otras palabras podemos desarrollar un programa sin saber programar.

Para iniciar con esta sección, tenemos que conocer por lo menos algunos conceptos que se manejaran dentro de la aplicación.

MENU - Grupo de opciones para hacer una selección.

MENU BAR - Menú de selección en forma HORIZONTAL.

MENU POP-UP - Menú de selección en forma VERTICAL.

STRUCTURE LIST - Menú POP-UP con nombres de bases de datos.

BATCH PROGRAM - Programa de ejecución por LOTES o grupos.

BANNER - Mensajes para dar la presentación del programa.

ITEM - Es cada una de las opciones de un MENÚ.

HELP TEXT - Pantalla alterna con texto de ayuda.

EMBED CODE - Incrustar código.

LAYOUT - Disposición, Presentación, Composición, Diseño, Trazado.

I.4.12 ALGORÍTMO GENERAL PARA GENERAR APLICACIONES

1.- EL LAYOUT.

Diseñar un LAYOUT (en papel) de la aplicación a generar. Para realizar un LAYOUT haremos un esquema de cómo queremos colocar el MENÚ principal (BAR) y cada uno de sus ITEMS, así como cada uno de los MENÚS POP-UP y sus ITEMS (No olvidar la SALIDA). Y por último, darle un nombre a la APLICACIÓN, a cada MENÚ BAR y POP-UP.

2.- LA DEFINICIÓN.

Definir datos de la APLICACIÓN. Pulsar la tecla enter en CREATE de APPLICATIONS, seleccionando APLICATION GENERATOR, y proporcionar los datos que aquí se requieren (tomar los nombres de nuestro LAYUOT) y salvar.

3.- EL BANNER.

Definir los letreros del BANNER de entrada bienvenida y (c).

4.- DISPLAY SING ON BANNER YES / NO

Pulsar Alt + A (Aplication) para desplegar o no este banner al momento de ejecutar la aplicación.

5.- DISEÑAR MENU BAR PRINCIPAL

Pulsar Alt (DESIGN) para diseñar el MENU BAR principal creándolo con el nombre de nuestro LAYOUT. Escribir los ITEMS de este menú pulsar F5 para iniciar cada ITEM y pulsar para cerrar cada ítem. Ajustar tamaño y posición del MENÚ BAR.

NOTA GENERAL:

- Es importante mencionar que para salir de algún menú de opciones DBASE podemos usar F10 y salvar o Ctrl + W. Pero si salvamos cuando estamos en alguno de nuestros marcos del menú o banner, DBASE salvará en ese momento todo lo que hemos diseñado y nos regresará al CONTROL CENTER. Para continuar tendremos que entrar nuevamente en nuestra aplicación y seleccionar mediante la opción DESIGN cada uno de los menús que habíamos creado.

- Cuando en una caja de DBASE nos solicite el nombre de un archivo podemos ayudarnos con SHIFT F1 y seleccionar ya sea una BASE DE DATOS, QUERIE, FORM, REPORT, LABEL o algún MENU previamente salvado.

6.- ITEM (bar) - OVERWRITE ASSIGNATED DATA BASE

Si alguno de los ITEMS requiere el uso de otra base de datos o QUERIE sobreescribir la base actual usando Alt y la opción OVERWRITE ASIGNATED DATA BASE, y seleccionar con barra espaciadora (above, BELOW , in efect at runtime), escribiendo a continuación la base (.DBF) o querie (.QBE) que se desea usar, así como el índice y orden, si es que los tiene.

7.- ITEM (bar) - CHANGE ACTION - OPEN A MENU

Cambiar la acción a cada ITEM con Alt seleccionando CHANGE ACTION y después OPEN A MENU, para que active el menú POP-UP que le corresponde, dando los nombres de nuestro LAYOUT.

8.- DISEÑAR LOS MENÚS POP-UP

Diseñar con Alt cada uno de los MENUS POP-UP con el nombre de nuestro LAYOUT, escribiendo los ITEMS en cada renglón, ajustando tamaño y posición de menú POP-UP.

No debemos olvidar el menú de salidas a DBASE y DOS, dejando para éste, TRES opciones y en la de EN MEDIO usar BY PASS ITEM ON CONDITION, esto con el fin de evitar una selección errónea por diferencia óptica de colores.

9.- MENÚ POP-UP e ITEMS
OVERWRITE ASIGNATED DATA BASE

Verificar que cada MENÚ POP-UP y cada ITEM active la BASE o QUERIE que le corresponde (barra de status) y de ser necesario, cambiarla con la opción sobreescribir *(OVERWRITE ASSIGNATED DATA BASE)* en Alt + M (Menú) o Alt (Item) según sea el caso.

VER PUNTO 6.

10.- ITEM (pop-up) - CHANGE ACTION

Cambiar la acción a cada uno de los ITEMS con Alt seleccionando CHANGE ACTION e identificar las acciones a realizar por cada ITEM :

Ver tabla de ejemplo siguiente.

° ITEM (opcion)	° ACCION	° ARCHIVO	° MODO
ALTAS	add, edit, delete	X.FMT	APPEND
BAJAS	Perform File Operation		Delete
CAMBIOS	add, edit, delete	X.FMT	EDIT
CONSULTAS	browse	X.FMT	NOEDIT
REORDENAR	Perform File Operation		REINDEX
ELIMINAR	Perform File Operation		PACK

REPORTES	Display or Print	X.FRM	REPORT
ETIQUETAS	Display or Print	X.LBL	LABEL
SUB-MENUS	OPEN A MENU		
----------	by pass item on condition		
ACTUALIZAR	DO Program dBase		
RESPALDO	Run Program DOS		BackUp
SALIDA	EXIT		RETURN TO CALL

Éstos son sólo algunos de los ejemplos que comunmente utilizamos en una aplicación, pero existen muchas más acciones que podemos seleccionar para cada ITEM.

11.- HELP TEXT.

Una de las características que todos nuestros usuarios toman ya como un estandar es la tecla F1 AYUDA, que les permite saber que es lo que tienen que hacer en alguna opción o pantalla de captura. Pues bien, no debemos permitir que nuestra aplicación adolezca de esta utilidad. Podemos adicionar tanto a los MENÚS como a cada ITEM de un texto de ayuda con Alt + M o Alt + I (menú-ítem) según sea el caso y la opción *WRITE HELP TEXT*.

12. INCRUSTAR CÓDIGO.

(Sólo para programadores de 3a generación)

Es posible que a cada ITEM le incrustemos una sección de código que deseamos se ejecute, ya sea antes o después de la acción del ITEM, pero para esto, es necesario conocer el lenguaje de DBASE 3a. generación y estar seguros de la sintaxis y lógica aquí escritos.

13.- GENERAR CÓDIGO.

Una vez que hemos cambiado la acción a cada ITEM, podemos ahora generar programa con Alt G y si lo deseamos podemos usar la opción *DISPLAY DURING GENERATION / YES* , Para ver a través de una ventana el código que se genera de nuestra aplicación al momento de pulsar *BEGIN GENERATION.*

Una aplicación con... UN MENÚ BAR, 2 POP-UP de bases de datos con altas, bajas, cambios, reportes y etiquetas, y un POP-UP de salidas a DBASE y DOS genera más o menos 1600 líneas de código, podemos tomar esta experiencia como patrón para saber si de nuestra aplicación se generó un tamaño de código equivalente.

14.- SALVAR, SALIR .

En teoría podemos decir que la aplicación ya funciona y está lista para usarse.

15.- EJECUTAR (correr).

Para poder hacer uso de nuestra aplicación, sólo tenemos que pulsar ENTER en el nombre de nuestra aplicación en el CONTROL CENTER, y no se extrañe, pero si, DBASE IV es el único programa que pregunta, " estas seguro de correr la aplicación.? "

NOTA :

Si algo no resulta como lo esperábamos, deberemos corregir a través del APPLICATION GENERATOR y volver a generar el código hasta cumplir con el objetivo de la aplicación.

Es posible editar el código que nos ha entregado DBASE, pero se pierde la relación con generador de aplicaciones. Y si se volviera a generar el código usando el generador de aplicaciones, los cambios realizados por EDICIÓN se perderán por completo.

Las extensiones relacionadas con una aplicación son :

XP.APP Es la aplicación diseñada en pantalla

XP.PRG Contiene la programación editable en código DBASE

XP.DBO Programa objeto de ejecución

XM.BAR Menú BAR

XNP.POP Una para cada Menú POP-UP

I.5 VENTAJAS, DESVENTAJAS Y UNA ALTERNATIVA.

Esperando que la metodología anterior sea de utilidad en los momentos en que la automatización de un problema es urgente, analicemos ahora ventajas y desventajas de esta herramienta, así como una alternativa utilizando este mismo software.

Podemos mencionar que las ventajas que obtenemos con dicho método son :

1.- Generación automatizada de la codificación de un sistema, sin conocimiento de un lenguaje tradicional de programación.

2.- Facilidad de organización y diseño de sistemas.

3.- Facilidad en la creación de formatos entrada de datos.

4.- Facilidad en la creación de reportes y dispositivos de salida.

5.- Tiempo reducido en el desarrollo del sistema.

Las desventajas del método anterior, a pesar de ser una forma muy rápida de desarrollar sistemas, son operativamente notorias al momento de utilizar el producto terminado (PROGRAMA.DBO).

Veamos ahora dos desventajas notorias.

1.- Código no optimizado (TAMAÑO).

Se basa en un modelo preestablecido y no en necesidades reales.

2.- Algoritmo pausado de cambio de menús (VELOCIDAD).

Por ser basado en un solo modelo el cambio de un menú POP-UP a otro, tiene que pasar por el menú BAR, y no directamente al menú pop-up que se desea.

Estas desventajas van de la mano con las ventajas 1,2 y 5.

Una alternativa a este método sin dejar a un lado las ventajas de los menús que dBase IV nos ofrece (BAR y POP-UP), consiste en crear un código reducido y que podamos acoplar a cualquier aplicación.

Empezaremos por presentar las principales estructuras de programación utilizadas en los lenguajes Xbase.

1.6 ESTRUCTURAS DE PROGRAMACION XBASE

Nota : La indicación de cuatro puntos (....) representa, una o varias acciones, comandos, asignaciones, funciones, rutinas, subprogramas, etc.

La estructura IF se utiliza para tomar decisiones lógicas, las instrucción ELSE, es una opción para otras acciones cuando la condición no se cumpla.

```
IF condición
  ....
ELSE
  ....
ENDIF
```

La estructura CASE se utiliza para tomar decisiones en base a una condicion múltiple, la instrucción OTHERWISE tiene el mismo uso que la instrucción ELSE anterior.

```
DO CASE
  CASE condición
    ....
  CASE condición
    ....
  CASE condición
    ....
  OTHERWISE
    ....
ENDCASE
```

La estructura DO WHILE se utiliza para hacer ciclos repetitivos mientras que una condición se cumpla, se pueden utilizar las intrucciones:

LOOP : Realiza un salto al inicio del ciclo DO WHILE.

EXIT : Realiza un salto a la instrucción fin de ciclo ENDDO.

```
DO WHILE condición
  ....
 [ EXIT ]
  ....
 [ LOOP ]
  ....
ENDDO
```

La estructura SCAN es igual al DO WHILE como si la condición de esta fuera EOF() " fin de archivo ".

```
SCAN
  ....
 [ EXIT ]
  ....
 [ LOOP ]
  ....
ENDDO
```

I.7. CONCLUSIONES DEL CAPÍTULO.

Como sugerencia : Revisar ahora en APENDICE A el programa titulado EJEMPLO1.PRG que es una propuesta a la alternativa tomada.

Nota : El código aquí incorporado sugiere un conocimiento básico del lenguaje xBase. Sólo tenemos que teclearlo y correrlo dentro de dBase IV para ver su uso y resultados.

Este programa es muestra de un controlador de persianas y es de autoenseñanza.

CAPÍTULO II

COMPILADORES PARA BASES DE DATOS

TEMAS DEL CAPÍTULO.

1.- INTRODUCCIÓN.

2.- COMPILADOR CONTRA INTÉRPRETE.

3.- PRODUCTOS COMPILADORES.

4.- CARACTERÍSTICAS DE LOS COMPILADORES.

5.- CARACTERÍSTICAS DEL LENGUAJE.

6.- ESTRUCTURAS DE PROGRAMACIÓN.

7.- COMPILACIÓN.

8.- COMPILACIONES PARCIALES DINÁMICAS.

9.--CONCLUSIONES DEL CAPÍTULO.

II.1.- INTRODUCCIÓN

En este capítulo analizaremos los compiladores para lenguajes de bases de datos y las características que cada uno de ellos presenta. Así como ventajas y desventajas al usar uno u otro.

Empezaremos por hacer una pequeña explicación de los términos que utilizaremos :

x-BASE :

Son todos aquellos productos que están enfocados al manejo de bases de datos y cada uno cuenta con un lenguaje muy parecido entre sí. Basado en los lenguajes PASCAL - COBOL y a últimas fechas incrementado con lenguaje C través de CODE BASE.

MODULAR :

Conjunto de programas interrelacionados jerárquica o independientemente a un programa principal.

LENGUAJE :

Conjunto sistemático de signos que permiten la comunicación y construcción de programas con los que se puede operar una computadora.

INTÉRPRETE :

Un intérprete es un programa desarrollado para traducir una a una las instrucciones de algún lenguaje de alto nivel al lenguaje de máquina, y de este modo realizar la operación de ésta instrucción.

Estos programas reciben el nombre de INTÉRPRETES porque esa es precisamente su labor, interpretar una instrucción y ejecutarla sin importar si las siguientes instrucciones del programa son erróneas.

Ejemplo : Si utilizáramos los servicios de una PERSONA TRADUCTOR para entendernos con otra persona que hablara un idioma diferente al nuestro, el traductor interpreta nuestras palabras y las traduce al idioma de la otra persona, sin conocer todo lo que vamos a decir más adelante ni el objetivo de nuestra plática.

COMPILADOR.

Un compilador es un programa desarrollado con el objetivo de traducir un bloque de instrucciones en lenguaje de alto nivel a lenguaje de máquina, ejecutando todas las instrucciones hasta que revisa todo el texto.

Estos programas reciben el nombre de COMPILADORES por que su objetivo es tomar todo el texto de un programa y traducirlo a código maquina obteniendo de esta operación un nuevo archivo, que puede ser (*.OBJ *.COM *.EXE ... ETC.).

Ejemplo : Podemos hacer una comparación entre el trabajo de un arqueólogo y un compilador, cuando el Arqueólogo tiene que traducir un pergamino de símbolos, el arqueólogo nos dirá lo que dice el pergamino hasta revisarlo completamente y comprender el contexto del mismo.

COMPILAR : Es el proceso para traducir un programa completo al lenguaje de máquina.

LIGAR : Es el proceso para convertir un programa compilado en un programa ejecutable.

II.2.- INTÉRPRETE Y COMPILADOR

¿ Cómo podemos decidir si usamos lenguaje interpretado o un lenguaje compilado ?. La decisión esta dada por los resultados que deseamos obtener :

Si queremos hacer pruebas en forma interactiva con el objetivo de un comando o una función, o el resultado de un nuevo algoritmo, si estamos empezando en la práctica de la programación. Lo más recomendable para estos fines sera un INTÉRPRETE.

Y cuando nuestras pruebas interpretadas, o los nuevos algoritmos dan buenos resultados, y queremos obtener la máxima velocidad de ejecución de nuestros programas, sera el momento de usar un COMPILADOR.

NOTA :

En modo interactivo los resultados de un programa INTERPRETADO son más rápidos por las características del mismo, por otro lado el COMPILADOR se toma procesos mas largos y complejos para entregarnos un programa resultante, que después debe ser ejecutado por separado, para obtener resultados.

Pero si esto mismo lo analizamos en un contexto global, el tiempo de respuesta del programa YA COMPILADO sera mucho menor que el programa a ser interpretado nuevamente.

Existen algunos lenguajes que se pueden usar como intérpretes o como compiladores, por ejemplo tenemos BASIC, PASCAL o FORTRAN.

Es importante mencionar que el lenguaje x-BASE se basa inicialmente en un interprete pero su mayor explotación se obtiene al ser compilado.

Así pues esperando que se tenga una idea mas clara de los términos compilador e intérprete haremos una reseña de los grupos de errores que encontramos comunmente al programar :

1 - ERRORES DE SINTAXIS.

2 - ERRORES DE SEMÁNTICA.

3 - ERRORES LEXICOGRÁFICOS.

4 - ERRORES DE LÓGICA.

1.- ERRORES DE SINTAXIS : Son los errores que encontramos al escribir un programa y alguna palabra no es conocida por el lenguaje.

2.- ERRORES DE SEMÁNTICA : Son los errores que encontramos cuando todas la palabras del programa son conocidas por el lenguaje, pero no tienen un orden o secuencia lógico.

3.- ERRORES LEXICOGRÁFICOS : Se refiere a que cuando escribimos el programa utilizamos caractéres no reconocidos por el lenguaje.

4.- ERRORES DE LÓGICA : Son los errores que el lenguaje no detecta al momento de revisar las instrucciones del programa, y como consecuencia los más difíciles de detectar.

Si el lenguaje detecta alguno de los tres primeros tipos de error enviara el mensaje : SINTAX ERROR. Pero un programa podra entregar resultados aunque en su contexto existan errores de lógica.

II.3.- PRODUCTOS COMPILADORES

Veamos ahora quiénes son los productos compiladores de bases de datos comercialmente más utilizados.

CLIPPER AUTUMN 86

CLIPPER SUMMER 87

CLIPPER 5.01

CLIPPER 5.2

DBASE IV 2.0

FORCE

FOXPRO 2.0

FOXPRO 2.5 FOR DOS

FOXPRO 2.5 FOR WINDOWS

CLIPPER AUTUMN'86

Representa el producto comercial que dio la primicia a compilaciones para programas dBase, es liberado en otoño de 1986 y respetaba las características del lenguaje dBase III plus, aunque se tenía cierta incompatibilidad con algunos comandos y el código original tenía que ser modificado.

CLIPPER SUMMER'87

Fue el producto preferido por la mayoría de los desarrolladores de aplicaciones x-Base, ya que respetaba completamente el lenguaje dBase III plus, con excepción de algunos comandos de

punto que listaremos más adelante, pero incluye una gran variedad de nuevos comandos y funciones que ofrecían mayores ventajas, como :

1.- Compilador y enlazador que genera programas ejecutables.

2.- Funcionamiento en redes de área local.

3.- Inclusión de procedimientos dentro del mismo programa (.PRG).

4.- Manejo de archivos DOS de bajo nivel.

5.- Funciones definidas por el usuario UDF sin límites.

6.- Posibilidad de 8 relaciones padre-hijo.

7.- Interfase para programas en C o Ensamblador.

8.- Funciones mejoradas para tratamiento de campos MEMO.

9.- Arrays unidimensionales y funciones para manejo de los mismos.

10.- Menús de tipo POP-UP.

11.- Activado y desactivado del cursor.

12.- Compilación con programas overlay.

CLIPPER 5.01

Es hoy uno de los productos de mayor explotación gracias a las ventajas que ofrece, además de todo lo anterior ya expuesto, incluye una nueva y larga lista de comandos y funciones para dar mayor fuerza al programador. Tal vez algo de lo más resaltante de esta versión es :

1.- Compilación de overlays dinámicos dentro del mismo (.EXE).

2.- Manejo de memoria Virtual (El .EXE pude ser mayor a 640 Kb).

3.- Pre-compilador o Compilador parcial (Mayor velocidad).

5.- Los programas (.EXE) son relativamente más rápidos que en otros productos.

6.- Las librerías de indexación.

7.- Las librerías que ofrece Nantuket TOOLS I y II.

8.- Programación Orientada a Objetos.

9.- Nueva escritura similar a lenguaje C, respetando el x-BASE.

CLIPPER 5.2

Ofrece la corrección de BUGS de la versión anterior y es ahora distribuido por la compañía C.A. (Computers Associated). además de utilizar reorganización en las librerías anteriores, que entre otras en lugar de Nantuket TOOLS se usa el CA-TOOLS.

= = COMANDOS DBASE NO SOPORTADOS POR CLIPPER = =

APPEND - ASSIST - CHANGE - CLEAR FIELDS - CREATE - DISPLAY - EDIT - EXPORT - HELP - IMPORT - INSERT - LIST FIELDS - LIST HISTORY - LIST STRUCTURE - LOAD - LOGOUT - MODIFY - MODIFY COMMAND - MODIFY LABEL - MODIFY REPORT - MODIFY SCREEN - RESUME - RETRY - RETURN TO MASTER - SET CARRY - SET DEBUG - SET DOHISTORY - SET ECHO - SET ENCRYPTION - SET FIELDS - SET HEADING - SET HELP - SET HISTORY - SET MENUS - SET STATUS - SET STEP - SET TALK - SET TITLE - SET TYPEAHEAD - SET VIEW.

DBASE IV 2.0 COMPILER

Este producto presenta realmente una buena opción, sobre todo cuando queremos desarrollar aplicaciones urgentes que por lo mismo requieren de muy poco tiempo. Podemos aplicar a este producto todo lo mencionado en el capítulo I y obtener un archivo ejecutable mediante el dB IV compiler. La gran desventaja de este producto es el tamaño de los archivos (.EXE) que se generan, ya que una sencilla aplicación de manejo de una base de datos, ocupa un (.EXE) de aproximadamente 1.4 MB, lo que no constituye una optimización de nuestra aplicación, sin embargo el tiempo necesario para desarrollarlo, es muy reducido, sin temor a exagerar podemos decir que aproximadamente 1 hora, empezando de cero, aún sin conocimientos del lenguaje xBASE.

FOX-PRO 2.0 (Distribution KIT).

Este producto que ofrece entre otras características :

1.-Compilación dinámica.

2.-Compilación parcial.

3.- Manejo de memoria dinámica.

4.- Ventanas o cajas de diálogo ajustables por mouse.

5.-Botones de radio.

6.-Inclusión de termómetros en operaciones de bases de datos.

7.- Manejo y operación de MOUSE intrínseco al (.EXE).

Tiene como principal característica que es además de todo un ambientes de manejo de bases de datos, con una interfase muy cómoda y fácil para los usuarios.

La gran desventaja de este producto, es probablemente el tamaño de los programas (.EXE) que genera, ya que estos varian alrededor de 1Mb y son comparativamente más lentos que los programas hechos en sus competidores.

FOX-PRO 2.5 (D.O.S. o WINDOWS).

Ofrece al programador no solo la corrección de BUGS de la versión anterior sino también una gran variedad de nuevas funciones, pero la característica principal de este producto es que ofrece la alternativa de que un programa escrito en x-BASE puede ser ejecutado bajo ambiente Windows sin tener que realizar en él ningún cambio, y con la inclusión de los campos OLE (Object Linked Embed - Incrustación Ligada al Objeto). De aquí que se considere por el momento uno de los productos con mayores expectativas, considerando la gran tendencia hacia los ambientes GUI. (Graphic Users Interface).

II.4.-CARACTERÍSTICAS TÉCNICAS DE LOS COMPILADORES

Número máximo de registros por base de datos	1,000,000
Número máximo de caractéres por registro	(RAM)
Número máximo de campos por registro	(RAM)
Número máximo de caractéres por campo	32 Kb
Número de dígitos de precisión en operaciones de cálculo	18
Número máximo de caractéres en una clave de indexación	250
Número máximo de índices por área de trabajo	15
Número de variables de memoria	2,048
Tamaño máximo de una variable de memoria (cadena)	64 Kb
Número máximo de dígitos en una variable numérica	19
Número máximo de arrays	2,048

II.5.- CARACTERISTÍCAS DEL LENGUAJE XBASE

A continuación revisaremos las reglas sintácticas para la escritura de instrucciones de un programa en lenguaje x-BASE usando : COMENTARIOS, ASIGNACIONES, CONDICIONES, COMANDOS, FUNCIONES Y ESTRUCTURAS DE PROGRAMACION.

COMENTARIOS

Son utilizados para dar explicación de variables, instrucciones, procedimientos, etc.

Ejemplos...

* Un asterisco al inicio de una linea indica comentario.

* NOMBRE DEL PROGRAMA :

* PARA QUE SIRVE :

* BASES QUE UTILIZA :

* CUANDO Y QUE HORA : empezó y terminó.

* QUIÉN LO HIZO :

Para hacer un comentario sobre la misma línea se usa el doble caracter &&

Ejemplos...

```
SET BELL OFF    && Desactiva el sonido de la campana.
SELECT 2        && Activa el área de trabajo 2
USE PERSONAL    && Abre la base de datos personal
GO 78           && Cambia apuntador al registro 78
```

ASIGNACIONES :

Es el proceso de indicar la asignación de un valor a una variable.

Ejemplos...

a = 0

store space(10) to B,C,E,F

Cantidad = 56

Importe = Cantidad * Precio

Donde el precio puede ser el campo de un registro de la base de datos o una variable de memoria.

VARIABLES DE MEMORIA

Mv=0 && Variable Numérica de valor cero

Mv=5 && Variable Numérica de valor cinco

Mv=SPACE(10) && Variable Caracter de 10 espacios

Mv=" " && Variable Caracter de 10 espacios

Mv=' ' && Variable Caracter de 10 espacios

Mv=.T. && Variable lógica de valor verdadero

Mv=.F. && Variable lógica de valor falso

Mv=DATE() && Variable con fecha de sistema

Mv=CTOD(" / / ") && Variable fecha en blanco

Podemos notar que existen diferentes formas de asignar valores a una variable aún siendo del mismo tipo, y que se pueden usar para títulos y constantes las comillas dobles o sencillas, respetando solamente que si se usa doble al inicio se debe usar doble a final.

CONDICIONES :

Se establecen por la comparación de 2 o más elementos, obteniendo de esto un valor lógico : .T. o .F. pudiendo establecer como condición un resultado lógico.

Ejemplos... Preguntando : si?

A = 5

B = .T.

.NOT. EOF()

.T.

PRECIO = 1000 .OR. ETIQUETA = "ROJA" .AND. .NOT. EOF()

COMANDOS :

Son las instrucciones que podemos utilizar de cada lenguaje :

Ejemplos...

COMANDO [FOR *condición* *[alcance]* *[WHILE condición]*

COUNT FOR edad 20 NEXT 200 WHILE edad 30

LIST FOR edad 20 NEXT 200 WHILE edad 30

SUM FOR edad 20 NEXT 200 WHILE edad 30

DISPLAY FOR edad 20 NEXT 200 WHILE edad 30

FUNCIONES :

Son procesos internos del lenguaje que entregan un resultado de una operación lógica, numérica, caracter, fecha, etc. Y sintácticamente se usan con paréntesis (), dentro de estos puede ir el argumento de la función.

Ejemplos...

FOUND()

DBF(2)

LEN(nombre)

DATE()

LEFT(nombre,2)

OPERADORES

a) OPERADORES ARITMÉTICOS

suma +

resta -

división /

multiplicación *

exponenciación ** o ^

agrupadores ()

b) OPERADORES LÓGICOS

Y lógico .AND.

NO lógico .NOT.

NO lógico !

O lógico .OR.

agrupadores ()

c) OPERADORES DE RELACIÓN

menor que <

mayor que >

menor igual < =

mayor igual > =

diferente < >

diferente !

diferente #

igual =

exactamente igual = =

cadena incluida $

agrupadores ()

II.6.- ESTRUCTURAS Y CÍCLOS DE PROGRAMACIÓN

Los lenguajes x-BASE son lenguajes estructurados que facilitan por esto la programación modular, y sus estructuras fundamentales de programación se identifican en tres grupos:

A) Estructuras de selección

IF-ELSE-ENDIF

DO CASE-CASE-CASE-CASE-OTHERWISE-ENDCASE

b) Estructuras de repetición

DO WHILE-ENDDO

FOR-NEXT

SCAN-ENDSCAN

c) Cambio de control del programa

DO programa.prg - RETURN

DO - RETURN

RUN programa.exe o .com

! programa.exe o .com

CALL programa.obj o .bin

ESTRUCTURAS DE SELECCIÓN

IF *condición*

    instrucciones

ELSE

    instrucciones

ENDIF

DO CASE

  CASE *condición*

        instrucciones

  CASE *condición*

        instrucciones

  CASE *condición*

        instrucciones

  OTHERWISE

        instrucciones

ENDCASE

ESTRUCTURAS DE REPETICIóN

DO WHILE *condición*

instrucciones

[LOOP]

[EXIT]

[RETURN]

ENDD

FOR variable = Número TO Número STEP Número

instrucciones

NEXT

DO SCAN

instrucciones

ENDSCAN

CAMBIO DE CONTROL DE UN PROGRAMA

DO procedimiento - RETURN

DO programa.prg - RETURN

RUN programa.exe o .com

! programa.exe o .com

CALL programa.obj o .bin

II.7 COMPILACIÓN

Para compilar un programa en clipper son necesarios los siguientes elementos.

a) Un editor que genere código ASCII estandard.

SK (Sidekick)

SKPLUS (Sidekick plus)

QEDIT (Quick Editor)

EDIT (de MS-DOS)

TP (Turbo Pascal)

TB (Turbo Basic)

TC (Turbo C)

PE (Program Editor - de clipper)

WS (WordStar - No Document)

b) El compilador CLIPPER.EXE

c) Las librerías CLIPPER

CLIPPER.LIB

NANTUCKET TOOLS - NT250.LIB

DBFNDX .LIB

Además de las librerías existentes de modo comercial, los programas que nosotros mismos escribimos se pueden convertir en librerías para ser usadas ligandandolas como programas (.OBJ)

d) Un enlazador.

PLINK86 de clipper

LINK del MS-DOS

TLINK de turbo C

RTLINK de clipper 5.0

El proceso de compilación consiste en los siguientes tres pasos:

A) ESCRITURA (código fuente)

La escritura de un programa xBase debe realizarse en base a las reglas sintácticas, semánticas y lexicográficas del lenguaje, el programa tendrá la extensión (.PRG) y a este se le denomina código fuente.

B) COMPILAR (de fuente a objeto)

El proceso de compilación consiste en traducir las intrucciones del código fuente a un archivo objeto (.OBJ), que es entendible por cualquiera de los enlazadores antes mencionados. Este paso lo realiza el programa CLIPPER y la sintaxis básica es la siguiente :

```
clipper programa.prg
```

C) LIGAR (de objeto a ejecutable)

El objetivo de ligar o enlazar es asociar los archivos obtenidos por el compilador a un programa (.EXE) que contendrá las instrucciones en lenguaje de máquina, el cuál podemos utilizar desde el PROMPT de sistema operativo tecleando sólo su nombre. La sintaxis básica para enlazar es :

```
rtlink fi programa lib nt250,dbfndx
```

** Esto en el caso de usar clipper 5.0 con las librerías de Nantucket Tools II y los controladores para archivos (.NDX).

Resumiendo: Una forma sencilla para agilizar estos pasos es generar un programa por lotes, que podemos llamar (1.BAT) y que está basado en el programa cl.bat incluido en clipper. Veamos el contenido de éste.

```
CLS
clipper %1
if not errorlevel 1 rtlink fi %1 lib nt250,dbfndx
if not errorlevel 1 %1
```

Con estas cuatro líneas del programa (1.BAT) y puesto este en ruta "PATH" podemos compilar fácilmente cualquier programa xBase sin importar el subdirectorio donde estemos trabajando.

Nota : Es importante resaltar que para el proceso de compilación existen muchas variantes que no se han mencionado, y para poder aprovecharlas es recomendable recurrir a los manuales de clipper ya sea Summer'87 o 5.0.

II.8.- COMPILACIONES PARCIALES DINÁMICAS

(CPD)

Una compilación parcial dinámica consiste en aprovechar las ventajas del compilador clipper 5.0 para pre-compilar parcialmente cada módulo de nuestro sistema. El objetivo de esto es ahorrar tiempo en el proceso de depuración o mantenimiento de sistemas multimodulares. Con el modo de compilación mencionado y usado por el (1.BAT) tenemos que recompilar "TODOS" los .PRG si es que en alguno de ellos hacemos modificaciones. Con el método que mencionaremos ahora optimizaremos tiempo ya que si algún módulo del sistema tiene que ser modificado "SOLO ESTE" sera recompilado ahorrándonos con esto tiempos importantes en la compilación.

Para poder pre-compilar los módulos de un sistema es necesario crear 3 módulos de instrucciones (no xBase), lo cual nos tomará un poco de tiempo extra.

1) CPD.BAT

2) CPD.RMK

3) CPD.LNK

Por lo tanto si el sistema que estamos desarrollando es demasiado grande y modularizado, el pre-compilador es una excelente opción.

Ejemplo esquematizado de un sistema modular:

COSTOS

MATERIALES	HERRAMIENTAS	PERSONAL	INDIRECTOS
ALTAS	ALTAS	ALTAS	ALTAS
BAJAS	BAJAS	BAJAS	BAJAS
CAMBIOS	CAMBIOS	CAMBIOS	CAMBIOS
REPORTES	REPORTES	REPORTES	REPORTES

A continuación revisaremos un ejemplo de CPD con CLIPPER 5.0 basado en el esquema anterior se listarán los 3 archivos necesarios : CPD.BAT - CPD.RMK - CPD.LNK

•• CPD.BAT ••••••••••••••••••••••••••••

COSTOS.rmk

if not errorlevel 1 COSTOS

••

```
.................................. COSTOS.RMK ..................................

.prg.obj:

  clipper $* /m

.obj.exe:

  rtlink @COSTOS

COSTOS.OBJ   : COSTOS.PRG

PROCEDUR.OBJ  : PROCEDURE.PRG

MATERIAL.OBJ  : MATERIAL.PRG

HERRAMIE.OBJ  : HERRAMIE.PRG

PERSONAL.OBJ  : PERSONAL.PRG

INDIRECT.OBJ  : INDIRECT.PRG


COSTOS.EXE: COSTOS.OBJ PROCEDURE.OBJ MATERIAL.OBJ HERRAMIE.OBJ \

PERSONAL.OBJ INDIRECT.OBJ

.....................................................................
```

COSTOS.LNK

```
# COSTOS.LNK
# Creación de COSTOS.EXE - Clipper 5.01 / Rtlink
# Versión 1.0
DYNAMIC INTO COSTOS.ov1
FILE COSTOS.obj
FILE PROCEDUR.OBJ
FILE MATERIAL.obj
FILE HERRAMIN.obj
FILE PERSONAL.obj
FILE INDIRECT.obj
FILE nt2us50.obj
# FILE errorsys.obj
BEGINAREA
  SECTION INTO COSTOS.ov2
  FILE M_ALTAS.OBJ
  FILE M_BAJAS.OBJ
  FILE M_CAMBIOS.OBJ
  FILE M_REPORTE.OBJ
  FILE H_ALTAS.OBJ
  FILE H_BAJAS.OBJ
  FILE H_CAMBIOS.OBJ
  FILE H_REPORTE.OBJ
  FILE P_ALTAS.OBJ
  FILE P_BAJAS.OBJ
  FILE P_CAMBIOS.OBJ
  FILE P_REPORTE.OBJ
  FILE I_ALTAS.OBJ
  FILE I_BAJAS.OBJ
  FILE I_CAMBIOS.OBJ
  FILE I_REPORTE.OBJ
ENDAREA
LIB nt250.lib
LIB dbfndx.lib
OUTPUT COSTOS
RELOAD FAR 200
# incremental  CUANDO YA EXISTE  EXE
# VERBOSE
```

II.9 CONCLUSIONES DEL CAPÍTULO:

Al hablar de los productos de compilación de xBase, tengo una fuerte inclinación por CLIPPER ya que me ha dado muchas facilidades de desarrollo, las compilaciones parciales dinámicas, la libertad y fexibilidad de lenguaje y tal vez lo más importante sea la seguridad de datos y la inter actuación entre desarrollo y aplicación, (Usando un editor residente como SIDEKICK).

Aúnque la balanza actualmente se inclina por FOXPRO/WINDOWS, por las tendencias actuales de nuestros usuarios hacia WINDOWS, no debemos perder de vista los requerimientos de hardware del mismo (Memoria RAM, el tamaño de los ejecutables generados en FOX PRO y el espacio que requiere en disco duro)

CAPÍTULO III

ANÁLISIS Y DISEÑO DE SISTEMAS

TEMAS DEL CAPÍTULO

1.- INTRODUCCIÓN.

2.- ETAPAS DEL CICLO DE VIDA DE UN SISTEMA (SENN).

2.1.- INVESTIGACIÓN PRELIMINAR.

2.2.- DETERMINACIÓN DE REQUERIMIENTOS.

2.3.- DESARROLLO DE UN SISTEMA PROTOTIPO.

2.4.- DISEÑO DEL SISTEMA.

2.5.- DESARROLLO DEL SOFTWARE.

2.5.1.- LENGUAJES DE PROGRAMACIÓN.

2.6.- PRUEBA DEL SISTEMA.

2.7.- PUESTA EN MARCHA.

3.- SEIS PASOS PARA CREAR UN SISTEMA (SANDERS).

3.1.- DEFINICIÓN DEL PROBLEMA.

3.2.- ANÁLISIS DEL SISTEMA.

3.3.- DISEÑO DEL SISTEMA.

3.4.- ANÁLISIS DE LA PROGRAMACIÓN.

3.5.- PREPARACIÓN DE PROGRAMAS

3.6.- INSTALACIÓN Y MANTENIMIENTO DE PROGRAMAS.

4.- DIAGRAMACIÓN.

5.- CONCLUSIONES DEL CAPÍTULO

III.1 INTRODUCCIÓN

El objetivo de este capítulo es proporcionar una síntesis conceptual de las etapas por las que pasa un sistema desde su planteamiento hasta la obtención de los frutos. En otras palabras, analizaremos lo correspondiente al ciclo de vida de un sistema.

El análisis es quizás la etapa más importante en el desarrollo de sistemas, ya que de ésta depende el buen o mal funcionamiento y las posibles mejoras o adecuaciones del sistema. De hecho si un sistema no es bien analizado, probablemente solo dará una solución parcial y temporal, sin la optimización de los recursos tiempo, costo, hardware y software.

ANÓLISIS Y DISEÑO :

En términos generales podemos decir que el análisis y diseño de sistemas es el proceso de estudio y revisión de acciones a ejecutar en el procesamiento de información, susceptibles de ser mejorados mediante el uso de nuevos métodos.

SISTEMA DE INFORMACIO:

Un sistema se puede definir como un conjunto de datos y programas relacionados entre sí, enfocados a obtener un fin u objetivo común.

III. 2 ETAPAS DEL CICLO DE VIDA DE UN SISTEMA

1.- Investigación preliminar.

2.- Determinación de requerimientos.

3.- Desarrollo de un sistema prototipo.

4.- Diseño del sistema.

5.- Desarrollo del software.

6.- Prueba del sistema.

7.- Puesta en marcha.

III.2.1.- INVESTIGACIÓN PRELIMINAR

La investigación preliminar consiste en :

A) Clarificación del requerimiento.

Este paso es nuestro primer contacto con el problema y muchas veces es resuelto con una simple entrevista al usuario, de lo que obtendremos como resultado el PLANTEAMIENTO DEL PROBLEMA.

B) Estudio de factibilidad.

El resultado de la clarificación del requerimiento nos dice si es factible la automatización del sistema, y para esto se involucran tres aspectos.

b.1.- Factibilidad técnica : Es la determinación de hardware, software y personas para dar solución al problema.

b.2.- Factibilidad económica : Consiste en analizar el costo estimado en base al tiempo de desarrollo así como el costo de operación y mantenimiento del mismo.

b.3.- Factibilidad operativa : Es cuestionar si una vez desarrollado y puesto en marcha los beneficios superan a la resistencia de los usuarios.

C) Aprobación de requerimiento.

En este paso podemos encontrar que no todos los proyectos son factibles por una u otra razón; o que probablemente nuestro usuario no requiere de un nuevo sistema, sino que para su problema ya existe por lo menos una solución. De lo contrario, esta aprobación es la conclusión del objetivo que perseguimos : No dejar duda entre lo que nos solicitan resolver y lo que resolveremos.

III.2.2.- DETERMINACION DE LOS REQUERIMIENTOS

Una vez realizada la investigación preliminar el punto clave del análisis de sistemas es conocer todas la facetas con las que estará involucrado el sistema dentro de la organización. Esto se logra por medio de una serie de cuestionamientos ya sea por medio de entrevistas, cuestionarios, o revisión de reglas y procedimientos dentro de la misma empresa.

"Y se deberá contestar a las siguientes preguntas :

1.- ¿ Qué se está haciendo ?

2.- ¿ Cómo se está haciendo ?

3.- ¿ Qué tan frecuentemente ocurre ?

4.- ¿ Qué tan grande es la cantidad de transacciones o decisiones?

5.- ¿ Qué tan bien se lleva a cabo la tarea ?

6.- ¿ Existe algún problema ?

7.- ¿ Si el problema existe, qué tan serio es ?

8.- ¿ Si el problema existe, cuál es la causa principal ?" [5]

Entrevistas :

Si el número de personas a entrevistar es muy grande podemos tomar sólo una muestra representativa de la población involucrada y realizar las entrevistas en forma individual o concertar una plática para que se contesten las preguntas en grupo.

Cuestionarios :

Cuando el problema no se logra aclarar por una entrevista o el número de personas es tan grande y/o de diferentes horarios que no se pueden reunir, deberemos aplicar un cuestionario para ser contestado por todo el personal y éste deberá ser revisado tal y como si de la persona misma se tratase.

III.2.3.- DESARROLLO DEL SISTEMA PROTOTIPO

En algunos casos no sera posible desarrollar un módelo o sistema prototipo, por no poder anticipar algunas de las tareas, pero de cualquier modo deberemos dar un planteamiento general del sistema que sera desarrollado, y aquellas tareas que aún no se logre anticipar su resultado y la relación con el proceso general, deberán ser tratadas cuidadosamente y por separado, nunca olvidarlas o menosprecialas.

El prototipo deberá ser un sistema diseñado para que pueda ser modificado con facilidad, la información recabada en el prototipo será la base para un sistema modificado y este será ahora el nuevo prototipo, repitiendose tantas veces como sea necesario para obtener las necesidades esenciales del diseño. Donde el desarrollo del prototipo coincide frecuentemente con el diseño final del sistema.

III.2.4.- DISEÑO DEL SISTEMA

Esta es la etapa base del éxito o fracaso del sistema, ya que aquí se debe hacer el planteamiento lógico del mismo. Es decir que se deberán plantear :

a) Los esquemas de salida de datos.

Son todos aquellos reportes que pueden ser direccionados a los dispositivos de salida como: Impresora, Pantalla, Archivo, etc.

b) Las estructuras que los almacenan o producen.

Archivos Aleatorios, Secuenciales, Bases de Datos o Fórmulas

c) Los formatos de entrada.

Formularios o Pantallas por medio de las que se recopila información.

d) Los cálculos o fórmulas.

Algoritmos matemáticos o lógicos para proceso de datos.

e) Los diagramas de árbol (WARNIER - JACKSON).

Son los diagramas usados para separar tareas por medio de llaves en forma de cuadros sinópticos.

f) Los diagramas de bloques.

Son usados para establecer las tareas que se irán realizando y la secuencia en el procesamiento de la información.

g) Los diagramas de flujo.

Son la representación esquemática basada en una simbología universal, y sirven para describir a detalle el proceso de un algoritmo que será aplicado a los datos para su transformación.

El diseñador del sistema es responsable de entregar todo esto a los programadores así como las especificaciones completas del sistema, y será éste quien se encargue de aclarar cualquier duda o confusión que exista cuando los programadores enfrenten las especificaciones del diseño.

III.2.5.- DESARROLLO DEL SOFTWARE

Este es el paso donde se decide en que y como deberán darse las instrucciones, en otras palabras, aquí decidimos en que lenguaje y bajo que técnica escribiremos el programa, para que sea lo más apropiado a la solución del sistema. El programador será responsable de la documentación del programa especificando claramente ¿ por qué ? y ¿para qué ?, utilizo cada procedimiento, variables, estructuras, etc.

Podemos hacer mención del objetivo fundamental para el que fueron creados algunos de los diferentes lenguajes :

Basic, Fortran, PL/1, Cobol, Rpg, Algol, Pascal, Ada, C, Logo, Ensamblador, Lisp, Logo, Prolog, Modula 2, Snobol, Xbase.

III.2.5.1 LENGUAJES

BASIC :

Beguiners All purpose Symbolic Instruction Code

Este lenguaje fue desarrollado para todo propósito y aunque esencialmente es enfocado a principiantes de la programación, con éste se pueden construir programas de mucha ayuda en la programación más avanzada. La versión más conocida fue creada por Bill Gates y como dato importante, con este GW-BASIC fundo la compañía MicroSoft.

FORTRAN :

Formula Translator

Este lenguaje fue desarrollado en 1954 patrocinado por IBM y dirigido por John Backus con el objetivo de facilitar las tareas de traducción de formulas matemáticas, sobre todo en problemas de ingeniería. La primera versión se uso en 1957 en la IBM-704, y el código del mismo tenía aproximadamente 25,000 líneas con un costo de programación de 2.5 millones de dolares.

COBOL :

Common Business Oriented Lenguage

Lenguaje desarrollado con propósito y orientación a los negocios y su primera versión se liberó en 1960 por CODASYL. Por la forma en que se escribe el lenguaje Cobol es más fácil para la gente de negocios que aún no programa, entender la lógica del código ya que usa palabras comunes dentro del área de negocios.

La estructura del lenguaje consta de las 4 divisiones siguientes :

- Identification Division, donde se hace la identificación general del programa.

- Enviroment Division, definición del ambiente de trabajo o equipo a usar.

- Data Division, su objetivo es presentar todos los elementos de información (archivos, registros, formatos y localidades.)

- Procedure Division, contiene el cuerpo principal del programa.

PL/1 :

Programming Lenguage / One

En los mismos años 1960's en que surge cobol, un comité de usuarios del IBM System/360 desarrolla un lenguage que intentaba ser universal, tanto para el área científica como para los negocios, tomando características de Fortran y de Cobol.

RPG :

Report Program Generator

Desarrollado en 1960 con el objetivo de facilitar la generación de reportes y actualización de archivos, aunque su campo es limitado por seguir un ciclo de procesamiento básico sin desviaciones, su ultima versión RPG-400 está basada en las máquinas IBM AS/400.

ALGOL :

Algorithmic Lenguage

Con el propósito de facilitar el paso de los algoritmos a un lenguaje estructurado fue diseñado Algol por un grupo de matemáticos y lo desarrollaron varios grupos en Europa y E.U.A., la última versión es Algol/68.

PASCAL :

En honor a Blas Pascal

Descendiente directo de Algol, desarrollado en la década 1960's por el profesor Nicklaus Writh en el Instituto Federal de Tecnología de Suiza. El Pascal fue el primer lenguaje importante que se creó, después de haberse difundido los conceptos asociados a la programación estructurada. Era un programa sencillo originalmente de facturación, similar a los que se presentan en otros lenguajes. Este lenguaje a tenido un gran auge en la última década con las versiones llamadas TURBO Pascal desarrolladas por Philip Khan colaborador del creador original y patrocinado por BORLAND. Actualmente la versión 7.0 es orientada a objetos.

ADA :

Lady Augusta Ada Lovelance (Colaboradora de Charles Babbage)

Considerada como la primer persona programadora del mundo.

Proviene en línea de Algol-Pascal, patrocinado por el Departamento de Defensa de los E.U.(DOD) para ser utilizado por las fuerzas armadas en 1975, para 1980 fue criticado por ser difícil de manejar e ineficiente, aunque sus partidarios lo califican como un gran avance en la tecnología de programación. Y como el DOD seguirá requiriendo computadoras militares que puedan usar ADA, el lenguaje continuará. Ada se convirtió a código estandard ANSI (American National Standards Institute) en 1983.

Lenguaje C :

Desarrollado por Bell Laboratories a principios de la década 1970 y usado para escribir una versión de UNIX, actualmente existen muchas versiones del lenguaje como son : Turbo C y C + + De las compañías Borland y MicroSoft.

Es el favorito de los programadores profesionales por ser un lenguaje de alto nivel y que permite a los programadores dar instrucciones en lenguaje de bajo nivel que se aproximan a las usada en lenguajes de máquina o ensambladores. En consecuencia el lenguaje C no es recomendado para personas que se inician en la programación.

LOGO :

A finales de la década 1960's Seymour Papert y sus colaboradores desarrollan LOGO, enfocado a problemas científicos serios, éste se popularizo como el primer lenguaje educativo que pueden usar los niños desarrollando habilidades tanto gráficas como matemáticas y con facilidades de sonido, ya que por medio de una punta de flecha o Delta o Tortuga permite dibujar, colorear y animar imágenes, así como el manejo de listas y cadenas, que puede también estar dirigido a la inteligencia artificial.

ENSAMBLADOR :

Durante la década 1950's se desarrollan códigos mnemotécnicos para las direcciones simbólica. Un mnemotécnico es una ayuda para la memorización y es mediante éstos que se facilita la escritura de programas en lenguajes numéricos de máquina a códigos representados por símbolos. De esta forma se origino una forma más fácil de comunicar a un programador con la computadora. Es importante hacer notar que el lenguaje ensamblador difiere según el microprocesador de la computadora donde se use.

LISP :

List Programming Lenguage - List Processor

Fue desarrollado por John McCarthy en 1959-1960 para apoyar las investigaciones en el campo de la Inteligencia Artificial, está enfocado a procesar datos y listas NO numéricas. Es el preferido de los programadores en este campo.

PROLOG :

Programación en Lógica

Lenguaje desarrollado en 1972 en la universidad de Marsella, Francia enfocado a la Inteligencia Artificial, fue la selección de un grupo japones para el desarrollo de sus computadoras de quinta generación. Y actualmente existe una versión Turbo Prolog, desarrollada por la compañía Borland

MODULA 2 :

Desarrollado por el inventor de Pascal Nicklaus Wirth, además de que conserva las ventajas de pascal es mas fácil de escribir y se considera que será más poderoso que Pascal. Por la semejanza entre ambos es relativamente fácil traducir un código escrito en Pascal a uno en Modula 2.

SNOBOL :

StriNg Oriented SymBOLic Language

Es un lenguage orientado al manejo de cadenas, utilizado para la manipulación de textos y recuperación de información, es una buena herramienta para los investigadores del área humanística.

xBASE :

x por todos los de BASES DE DATOS

Como ya lo hemos mencionado en el capítulo anterior es un lenguaje de programación estructurada basado en características de Pascal-Cobol con el objetivo fundamental de manipular Bases de Datos.

Una vez que se ha determinado cual será el lenguaje más adecuado para facilitar la escritura del programa se deberán hacer algunas preguntas como :

- ¿ Están familiarizados los programadores con el lenguaje ?
- ¿ Cual es la naturaleza de la aplicación ?
- ¿ Permite el lenguaje programación estructurada y/o modular ?
- ¿ Con que frecuencia se va a procesar la aplicación ?
- ¿ Se va a modificar a menudo el programa ?
- ¿ Se planea un cambio de equipo durante la vida del sistema ?
- ¿ Cuenta el lenguaje con apoyo ? (Actualizaciones y Mejoras)

Después de que se ha seleccionado tanto el lenguaje como el grupo de programadores que cumplen y satisfacen a los cuestionamientos anteriores, se procede a delegar la escritura del programa en secciones como :

a) Pantallas de captura.

b) Presentación de menús de selección.

c) Formatos de los reportes.

d) Algoritmos del proceso de Cálculo.

e) Seguridad de datos tanto en equipos monousuario como compartidos.

f) Pantallas de ayuda.

g) Interrupción de errores

En este proceso, el líder del proyecto determinara en base al diseño del sistema, la forma en que serán presentados los datos en pantalla y reportes, así como los colores y estilos de impresión, para que el sistema tenga una presentación uniforme, siendo de esta forma más fácil de operar por los usuarios.

III.2.6.- PRUEBA DEL SISTEMA

Este podrá ser el primer contacto de los usuarios con el sistema o bien puede ser probado por personal que no este involucrado en este proyecto y que por lo tanto requiere una capacitación del manejo del software, examinando si existen datos u operaciones que escaparon a las etapas anteriores para su re-estructuración adecuada.

Por lo tanto el sistema deberá ser lo bastante flexible para correcciones y/o adecuaciones futuras, Es importante tratar al sistema en forma imparcial, de hecho, como si nosotros mismos fuéramos los usuarios y desconocieramos al equipo de trabajo, solo así garantizaremos que el software será aprobado por nuestros usuarios, haciendo de esta forma un software más solido y confiable.

A la etapa de prueba del sistema debe darse especial atención y tiempo suficiente para revisar todas las partes del sistema, sin restar importancia a ningún dato, pantalla o procedimiento, llevando una bitácora de puntos a corregir, para con éstos, retroalimentar al equipo de programadores. En las siguientes pruebas se revisarán primero las anotaciones hechas en la bitácora y después otra revisión general, repitiendo esta prueba hasta estar seguros de que no existen errores.

III.2.7.- PUESTA EN MARCHA

Una vez concluidas y aprobadas todas las etapas anteriores será posible poner en marcha el sistema, dependiendo del tamaño de la empresa, el sistema y riesgo asociado. El responsable del sistema podrá instalar una versión beta o piloto solamente en una área de la compañía, para certificar su buen funcionamiento y más adelante, si no existen diferencias o errores entre lo esperado y lo entregado será puesta en marcha la versión definitiva.

III.3.- SEIS PASOS PARA CREAR UN SISTEMA

Revisando otras metodologías en el desarrollo de sistemas, encontraremos diferentes formas de tratar el Análisis y Diseño de Sistemas. Como ya hemos revisado el método de James A. Senn [5], veamos ahora los pasos que nos presenta Donal H. Sanders [1]

III.3.1.- Definición del problema:

Este paso es el primer contacto con las necesidades a resolver y consiste en identificar todas los factores que intervendran en el desarrollo del sistema, tales como : Factores tecnológicos, humanos, económicos y sociales, así como las fuentes de información que alimentarán al sistema y resultados que este deberá entregar.

III.3.2.- Análisis del sistema :

En esta etapa se reconoce que el nuevo sistema casi siempre soluciona un proceso ya existente, por lo que se deberá solicitar todos los elementos que se usan en colección de datos y a que resultados se quiere llegar.

Para esta recopilación de datos, se pueden utilizar algunas de las siguentes técnicas:

- Organigramas y normas de la organización.

- Diagramas de flujo de sistemas.

- Diagramas de flujo de datos.

- Diagramas de rejillas : Consiste en hacer una hoja tabular donde las columnas serán los reportes o salidas de datos del sistema, los renglones serán los formatos de entrada o captura. Y en las celdas o rejillas de intesección se marcará en que formatos de entrada estan los datos que nesecitan los reportes.

- Cuestionarios y formas especiales.

- Entrevistas y observaciones.

III.3.3.- Diseño del sistema:

Esta es la etapa en la que se determina la forma en la que será presentado el sistema, los algoritmos internos que utilizará y la forma en que entregará resultados. Para esto deberan considerarse :

- Planes a largo plazo.

- Flexibilidad.

- Precauciones de control y factores humanos.

- Crear un sistema o comprar paquetes comerciales que satisfagan las necesidades.

- Aspectos económicos.

Por último en esta etapa se deberá hacer una revisión del diseño y finalmente preparar un prototipo del sistema.

III.3.4.- Análisis de la programación:

La importancia del análisis de la programación determinara el lenguaje de programación que será utilizado para desarrollar el sistema, buscando que éste sea el más apropiado al tipo de problema que se desea solucionar. Asimismo se deberá determinar la metodología de programación que se utilizará. Para este paso existen en este momento una gran gama de alternativas.

- Programación Lineal

- Programación Estructurada

- Programación de listas y predicados (Inteligéncia Artificial)

- Herramientas CASE

- Programación por Macros

- Programación Orientada a Objetos

- Programación de Componentes

III.3.5.- Preparación de programas:

En esta etapa pueden intervenir uno o varios programadores que se encargarán de traducir las conclusiones obtenidas en las etapas anteriores al lenguaje y metodología seleccionado para desarrollar el sistema.

Al finalizar la escritura de programas y sus pruebas parciales, se deberán realizar varias pruebas globales al sistema para verificar que cumple con los requerimientos solicitados.

III.3.6.- Instalación y mantenimiento de los programas:

Una vez terminada la preparación de los programas y realizadas las pruebas globales se procede a la instalación del sistema, el cual podrá ser mejorado constantemente en base a nuevas necesidades que surgan de éste o de los usuarios, por lo que se deberá llevar un registro desde la primera versión y las subsecuentes modificaciones indicando las características generales de cada una de ellas con respecto a la anterior, asimismo los factores de riesgo de una versión a otra.

Como podemos observar los seis pasos aqui presentados presentan una gran similitud con el metodo anteriormente estudiado. Y por lo general, muchas de las diferentes formas de relizar el Análisis y Diseño de Sistemas incluyen etapas o pasos parecidos a los anteriores.

III.4.- DIAGRAMAS

Otro de los aspectos que debemos manejar es el tipo de digramas que se utilizan en el desarrollo de sistemas.

Diagramas de flujo - Es la presentación esquemática de la solución de un problema, indicando el flujo o secuencia que seguirán los datos desde la entrada del programa hasta la entrega de resultados. Estos diagramas se diseñan en base a un conjunto de símbolos que ilustran la operación que deseamos realizar.

Diagramas Nassi-Shneiderman - Estos diagramas obligan a realizar una programación estructurada (top-Down) y sin GOTOs, son conocidos tambien como diagramas de bloques o diagramas estructurales, y utilizan 4 símbolos básicos:

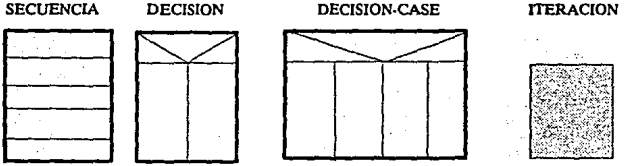
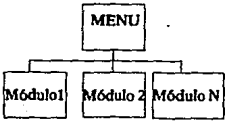


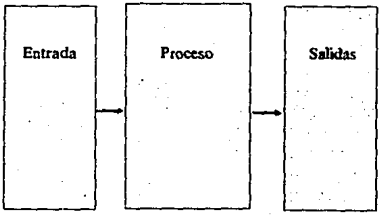
Diagrama de Jerarquía Más Entrada Proceso Salida (HIPO-*Hierarchy plus Input Proces Output*)

Desarrollados por IBM, es una técnica que se utiliza para mostrar las entradas, procesos y salidas asimismo sirve como herramienta para la documentación. Esta forma de diagramar implica la utilización de dos tipos de diagramas. 1)los diagramas de tabla de contenidos (VTOC), y 2)los diagramas funcionales que indican la secuencia de entrada - proceso - salida.

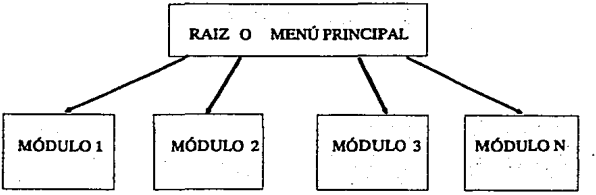
1) Tabla Visual De Contenidos



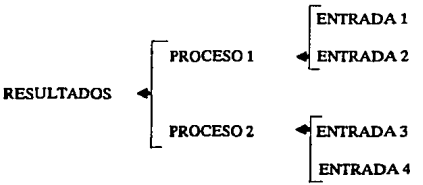
2)Diagrama Funcional



Diagramas de Estructura. Son utilizados para mostrar la estructura general del sistema, cada rectángulo representa un módulo de la aplicación. Son muy similares a los VTOC.



Diagramas de Warnier-Orr - (Jean Dominique Warnier) Desarrolló la técnica de construcción lógica de un programa y (Kenneth Orr) incluyó este método al diseño de sistemas, el diagrama se inicia con los procedimientos de salida y de ahí se parte para determinar los procesos y entradas que los producirán. Estos diagramas son basados en llaves.



III.5. CONCLUSIONES DEL CAPÓTULO

Existen diferentes metodologías para el análisis y diseño de sistemas entre las que podemos optar por una u otra según sea conveniente, es decir que cada persona puede escoger una metodoligía por que se le facilita más o porque la comprende mejor.

Es importante hacer notar que existen muchas otras formas de realizar el análisis y diseño de sistemas y que aún faltan muchas por descubrir, algunas de las nuevas formas serán en este momento la técnica característica de algún desarrollador.

Cuando seguimos alguno de estos métodos y el ANÁLISIS Y DISEÑO DEL SISTEMA se realiza correctamente podemos estar seguros que el sistema será confiable y se obtendrán del él los resultados esperados, haciendo las tareas complejas un poco o mucho más sencillas.

No hay que olvidar que así como las empresas irán cambiando y mejorando sus reglas y procedimientos, nuestros sistemas también son susceptibles de ser mejorados, por lo que debemos considerar el proceso de puesta en marcha como un proceso continuo.

Con relación a la metodología de movimiento (MOVE), podemos decir que cualquiera de los métodos de análisis y diseño de sistemas antes revisados puede ser aplicados, y se recomienda usar los diagramas de Nassi-Shneiderman, para representar cada uno de los procesos en un programa (MOVE).

CAPÍTULO IV

METODOLOGÍA DE MOVIMIENTO

TEMAS DEL CAPÍTULO

1.- INTRODUCCIÓN.

2.- ¿ POR QUÉ SURGIÓ ESTA IDEA ?.

3.- ORIGEN DE LA METODOLOGIA "MOVE".

4.- MODULOS Y PROBLEMAS.

5.- DIGRAMAS DE LA METODOLOGIA "MOVE".

6.- RESÚMEN.

7.- CONCLUSIONES DEL CAPITULO.

IV.1 INTRODUCCIÓN

La metodología MOVE consiste en asociar a cada base de datos los procedimientos necesarios para poder movernos entre los registros de forma natural, el objetivo es que con sólo usar las teclas de movimiento del cursor cambiemos el apuntador actual de registro, al mismo tiempo que se visualiza la información de éste, dando oportunidad de hacer correcciones, agregar registros, buscar información, marcar, desmarcar, suprimir registros marcados, cambiar de ordenamiento, mostrar la base de datos en forma tabular, imprimir, etc, asimismo todas las acciones que particularicen a cada base de datos.

IV.2 ¿ POR QUÉ SURGIÓ ESTA IDEA ?

Durante mi desempeño como Usuario, Operador, Desarrollador y Analista de Sistemas, en cada una de estas etapas encontré una gran resistencia a la operación de los sistemas por parte de los usuarios, operadores y capturistas, debido a que la mayoría de los sistemas de información carecen de una interfase natural de operación.

Por dar un ejemplo muy común: Es muy frecuente encontrar a un usuario que en el proceso de dar altas a una base de datos comete algúnos errores, ya sea en el registro actual o en registros anteriores, el proceso normal para corregir éstos errores sería:

- *Regresar al menú principal*
- *Seleccionar la opción Consultas o Cambios*
- *Teclear la clave o dato a buscar (experto en los datos)*
- *Corregir la equivocación*
- *Regresar al menú principal*
- *Seleccionar nuevamente la opción altas*
- *Continuar en la captura*

De ésta forma, además de hacer un trabajo lento por la cantidad de operaciones realizadas, el usuario tiene que ser experto en datos para localizar un registro sobre todo si el error esta en la llave de búsqueda.

IV.3 ORÍGEN DE LA METODOLOGÍA DE MOVIMIENTO

Si en lugar de sistemas por computadora los usuarios continuaran recopilando datos en formularios a papel y lápiz, como aún se realiza en algunas empresas, en el momento de escribir los datos en un formato, se necesita hacer alguna corrección ya sea en la hoja actual (registro) o en cualquiera de las hojas anteriores basta con mover el lápiz al dato (campo) y corregirlo y más aún, si el error está en las hojas anteriores basta con buscar con los dedos y la vista, la hoja con el error o simplemente la hoja que queremos revisar. Todo esto es posible sin cerrar el folder (archivo) y preguntarnos si ya no queremos seguir capturando, o conocer exactamente el número de hoja donde queremos consultar o corregir, lo cual nos convierte en expertos en los datos y si el experto en datos no se encuentra nadie podrá consultar los folders o archiveros donde se almacena la información.

Como consecuencia de esto, es mi intención aportar la metodolgía de movimiento; como una mezcla de los conocimientos aprendidos de mis maestros y mis usuarios. Asimismo una serie de procesos para simular el manejo natural de la información, con la aclaración de que si esta metodología "a mi manera de verlo", mejora el clásico menú de ABC (Altas Bajas Cambios) es también la misma, susceptible de ser mejorada.

De hecho, con gran sorpresa encontré en la presentación de FOX-PRO 2.5 for WINDOWS que como una característica nueva del producto, a cada Base de Datos generada en él se le asocia un procedimiento natural de manejo de la información. Junto con lo anterior viene la confirmación de que los usuarios necesitan cada vez más de interfases más cómodas y naturales para facilitar el control y manejo de datos al usar computadoras.

IV.4 MÓDULOS Y PROBLEMAS

A CONTINUACIÓN EXPONDREMOS UNA SERIE DE RUTINAS QUE DAN LAS BASES CONSECUENTES DE LA METODOLOGÍA DE MOVIMIENTO (MOVE):

ASCII, Teclado, Apuntadores, Pantalla base, Muestra, Captura, Búsquedas, Validación, Tablas de consulta, Unión de todas las partes, Datos dependientes, Optimización del método última versión.

Es importante mencionar que las rutinas aquí presentadas son basadas en xBase usando CLIPPER 5, para cualquier otro xBase deberán realizarse los cambios correspondientes de acuerdo al mismo.

Si se desea ver la solución de cada problema consulte el APÉNDICE *A* buscando el nombre del programa correspondiente al problema.

PROBLEMA 1

☐ *(TECLAS.PRG)*

Conocimiento de cada tecla y su posición en el código ASCII.

Diseñar un proceso que nos permita obtener toda la información de cualquier tecla presionada.

PROBLEMA 2

☐ *CURSOR.PRG*

Tener el control total del teclado:

Diseñar un procedimiento basado en los valores ASCII que nos permita mover un punto (caracter) en la pantalla hacia todos lados, ya sea pintando o borrando dicho caracter y usando las teclas de control de movimiento del cursor (Flechas).

PROBLEMA 3

☐ *ESTADO.PRG*

Identificar los límites de la base de datos.

Diseñar un procedimiento que guarde en variables de memoria (apuntadores) los límites de la base de datos en cuanto a posición de los registros INICIAL, FINAL y ACTUAL. Usando las teclas de movimiento de cursor, navegar entre los registros de modo en que nos parezca más natural, hacia adelante, atrás, inicio, fin, 10 o 20 registros adelante o atrás y salida, sin exceder los límites antes declarados.

PROBLEMA 4

Poner en pantalla los letreros y cuadros base para los datos.

Extraer de un formato de captura FMT las instrucciones de cuadros y letreros, es decir las líneas (@ say) " *NO ES UN PROGRAMA* "

PROBLEMA 5

Mostrar los datos del registro actual.

Extraer de un formato de captura FMT las instrucciones de captura de cada campo, es decir, las líneas (@ get) y convertir cada instrucción Get en Say. " *NO ES UN PROGRAMA* "

PROBLEMA 6

Mostrar los datos del registro actual y permitir su edición.

Extraer de un formato de captura FMT las instrucciones de captura de cada campo, es decir, las líneas (@ get) y añadir la instrucción READ. " *NO ES UN PROGRAMA* "

PROBLEMA 7

☐ *BUSCAR.PRG*

Buscar y encontrar información por llaves (Índices) múltiples.

Diseñar un procedimiento que identifique la llave de indexación actual, posicione el cursor en el campo o dato correspondiente permitiendo la captura por medio de éste y utilice los métodos LOCATE, SEEK o FIND, actualizando los apuntadores de la base.

PROBLEMA 8

☐ *VALIDAR.PRG*

Validar los datos capturados.

Diseñar una rutina que no permita duplicidad de información, esto sí y sólo sí el sistema lo requiere, así como conocer las funciones para validar datos por RANGOS, CADENAS.

PROBLEMA 9

☐ *TRAERDAT.PRG*

Tomar datos usando tablas de otra base de datos.

Diseñar un procedimiento que nos permita visualizar una tabla de posibles datos para ser puestos en el campo de captura actual, por ejemplo en una factura, el NOMBRE DEL CLIENTE de la base de datos CLIENTES por medio de la activación de una tecla de función o interrupción como : F9, F10 o Alt + X. Aceptando el dato con ENTER y dejando en blanco con Esc.

PROBLEMA 10

☐ *MOVE01.PRG*

Programa básico de movimiento dentro de una base de datos.

Utilizando los problemas anteriores combinarlos para obtener un solo programa que nos permita movernos en la base de datos mientras es presentada la información de cada registro, con la posibilidad de capturar o corregir en el momento que el usuario lo quiera.

PROBLEMA 11

☐ *PFPRMOVE.PRG*

Identificar los límites relativos de una base de datos.

Seleccionar y trabajar sólo con los datos dependientes.

Diseñar un procedimiento basado en los problemas 10 y 3 que nos permita generar apuntadores relativos de INICIO, FINAL y ACTUAL en los registros de una base de datos de nivel dependiente. Por ejemplo, identificar con apuntadores, sólo los productos que pertenecen al número de factura presente en pantalla. (Usar el siguiente modelo)

SELECT 1

USE FACTURAS

* Base con Encabezados y Totales de cada facturas

SELECT 2

USE FACTPROD

* Base con número de productos variable para cada factura

* Como dato general : El uso de FILTROS limita el control de los datos al mismo tiempo que hace más lento el desempeño del programa.

PROBLEMA 12

☐ *MOVE.PRG*

Optimizar los resultado de estos problemas en un procedimiento de uso general desligando los datos característicos de una sola Base de Datos para poder heredar este código a cualquier otra base.

PROBLEMA 13

☐ *MENUPERS.PRG*

Diseñar un programa de control de menús bajo un esquema PULL-DOWN, BAR-POPUP O PERSIANAS, Que nos permita manejar varios programas, (un MOVE para cada ventana)

NOTA IMPORTANTE: La mejor forma de adquirir estos conocimientos es, hacerlos propios mediante la solución y programación de cada uno de ellos, en base a ensayo y error.

(VER SOLUCIONES DEL CAPITULO EN EL APENDICE A)

IV.5. DIAGRAMAS DE LA METODOLOGÍA DE MOVIMIENTO

Utilizando las bases del análisis y diseño de sistemas usaremos las siguientes relaciones :

Diagramas de Flujo - Algoritmos de Proceso de datos

Diagramas de Bloques - Programas MOVE

Diagramas Nassi Shneirdeman - Programas MOVE

Diagrama de Estructura - Menú Principal (Pull-Doun)

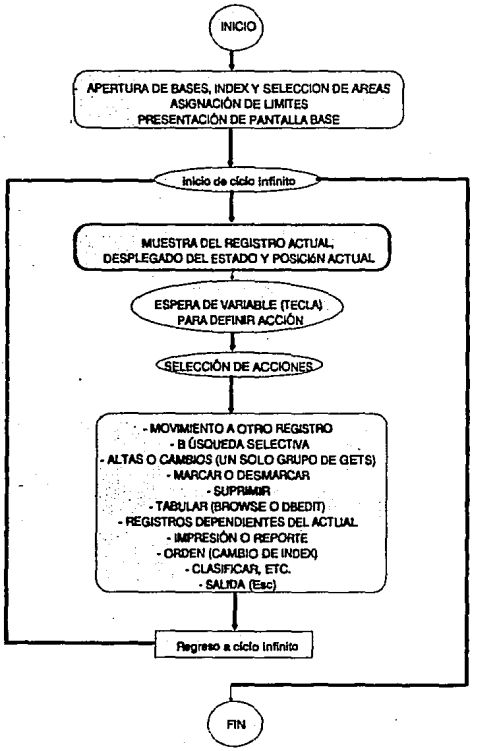
Los diagramas de flujo son utilizados, como ya se explicó, para describir detalladamente el proceso de modificación o cálculo de datos, y es muy variante para cada caso de procesamiento.

Utilizaremos un diagrama de bloques para representar los módulos y en general la metodología de movimiento (Ver el siguiente diagrama. "MOVE").

El diagrama de Estructura será utilizado. como ya se ha mencionado, para el menú principal del sistema. Así como en su momento presentamos (CAPÍTULO I) el modelo del menú principal para dBase IV y FoxPro, veamos ahora su equivalente para Clipper.

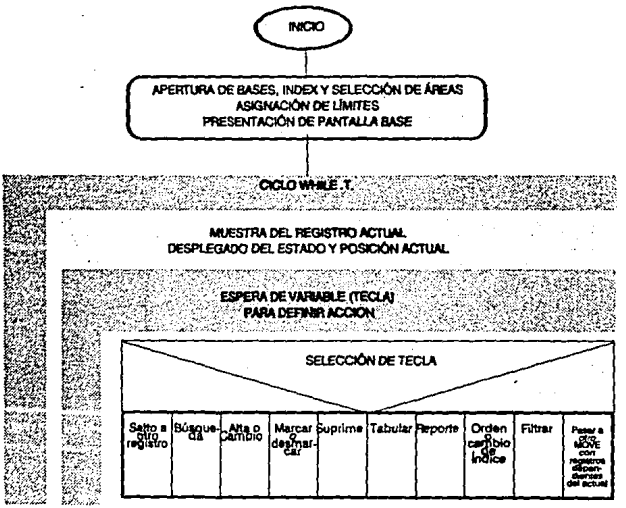
MOVE

REPRESENTACIÓN EN DIAGRAMA DE BLOQUES



MOVE

REPRESENTACIÓN EN DIAGRAMA NASSI-SHNEIDERMAN



IV.6. RESUMEN

Como parte de la adquisición de conocimientos intentemos diseñar (antes de ver la solución) un programa de propósito general para heredar a cualquier aplicación. Esté deberá ser basado en la instrucción (@ PROMPT) y SAVESCREEN (Ver el siguiente tip).

Para facilitar el diseño de este programa.

- Al estar dentro de un menú (@ PROMPT) las teclas flecha derecha y flecha izquierda, funcionan también como flecha abajo y flecha arriba simultáneamente, si nuestro objetivo es pasar de un menú (@ PROMPT) a otro debemos asignar dos procedimientos para cambiar la acción de estas teclas del siguiente modo :

La tecla flecha DERECHA debe convertirse en la secuencia:

Esc, Flecha derecha y Enter.

La tecla flecha IZQUIERDA debe convertirse en la secuencia:

Esc, Flecha Izquierda y Enter.

Para lograr esto veamos los siguientes procedimientos.

Al estar en el menú principal:

SET KEY 19 TO && Regresa valor original a la tecla flecha izquierda.

SET KEY 4 TO && Regresa valor original a la tecla flecha derecha.

Al estar en los sub-menús

SET KEY 19 TO *flecha_izq* && Asigna nueva función a la tecla.

SET KEY 4 TO *flecha_der* && Asigna nueva función a la tecla.

Después del Fin del programa o en procedimientos

```
*****************************************************************************

PROC flecha_izq

PARA pn, pl, rv

* return, up arrow, return

KEYBOARD chr(27) + chr(5) + chr(13)

RETURN

*****************************************************************************


*****************************************************************************

PROC flecha_der

PARA pn, pl, rv

* return, right arrow, return

KEYBOARD chr(27) + chr(4) + chr(13)

RETURN

*****************************************************************************
```

IV.7.- CONCLUSIONES DEL CAPÍTULO

Con esto facilitaremos la creación y control de menús para cualquier aplicación a desarrollar.

Ahora veamos en el APENDICE *A* algunos ejemplos que dan solución a los problemas anteriores, así como la escritura general del menú principal para cualquier aplicación *(MENUPERS.PRG)* y por último una serie de procedimientos incluidos en el archivo *(PPRO.PRG)* anexo en el diskette. Con el agradecimiento a mis compañeros desarrolladores, ya que estas rutinas son recopiladas de varias personas. Espero que logren ser de utilidad para quien en este momento utilice esta metodología, al mismo tiempo que podamos enriquecer cada vez más este conjunto de procedimientos, para optimizar TIEMPO y CODIGO, y en consecuencia, lograremos que nuestros programas ocupen menos recursos .

CAPÍTULO V

APLICACIÓN EN UN SISTEMA DE PRESUPUESTOS

TEMAS DEL CAPÍTULO

1.- INTRODUCCIÓN.

2.- INVESTIGACIÓN PRELIMINAR.

3.- DETERMINACIÓN DE LOS REQUERIMIENTOS.

4.- SISTEMA PROTOTIPO.

5.- DISEÑO DE UN SISTEMA PROTOTIPO.

6.- DESARROLLO DEL SOFTWARE.

7.- PRUEBA DEL SISTEMA.

8.- PUESTA EN MARCHA

9.- DESCRIPCIÓN GENERAL DE LOS PROGRAMAS PRINCIPALES.

10.- CONCLUSIONES DEL CAPÍTULO.

V.1. INTRODUCCIÓN

El sistema que se presenta en este capítulo está completamente realizado con los conceptos de la teoría de movimiento, de hecho este sistema ha pasado por varias etapas o versiones del programa MOVE, desde la primera versión (MOVE 1.0) para dBase IV, en la que el sistema era considerablemente lento además de varias pérdidas de información en campos MEMO, hasta la última versión (MOVE 6.0) para Clipper 5.0 donde se trabaja actualmente con un rápido desempeño y sin pérdidas de información.

Para lograr un sistema sólido se utilizaron los conocimientos escenciales del ANÁLISIS Y DISEÑO DE SISTEMAS de donde se desprendieron los resultados siguientes.

V.2. INVESTIGACIÓN PRELIMINAR

Al tener las primeras pláticas con la compañia que solicitaba la automatización del proceso, se obtuvo la siguiente explicación del problema:

- La elaboración de presupuestos se hacía basándose en un banco de tarjetones escritos a máquina, que contenían la descripción de cada uno de los productos manejados por esta empresa.

- La información en los tarjetones variaba tanto en estructura como en tamaño, cantidad de materiales usados, tipos de productos, dimensiones, uso, peso, y el tamaño de cada tarjeton iba desde 2 renglones hasta 3 cuartillas. Lo que nos obliga a pensar en campos de tipo MEMO.

- No existía catálogo de claves para productos o un nombre corto para referirse a ellos.

- Se utilizaba una ayuda de software (FRAMEWORK) pero solo cuando el presupuesto era pequeño en numero de partidas.

- El cálculo de importes ($) no presentaba un problema complejo, siempre y cuando el presupuesto no necesitara modificaciones en cuanto a : más o menos productos, recálculo por cambio de cotizacion del dólar, o algun otro cargo o descuento.

- Un presupuesto complejo y bien presentado ocupaba a varias personas durante un largo tiempo.

- Antes de que se intentara solucionar este problema usando la teoria (MOVE), se probaron algunos productos comerciales diseñados para administración empresarial. Pero por las características de la información no cumplían con los requisitos del cliente.

- Aunque el problema inmediato eran los presupuestos se tenía pensado a corto tiempo la automatización de pedidos, remisiones, facturación y órdenes de compra.

V.3. DETERMINACIÓN DE LOS REQUERIMIENTOS

- (Característica que debería cumplir)

- Lo principal del sistema es que soportara localidades de información de un tamaño suficientemente grande para las descripciones de los productos.

- Que permitiera tomarlos desde un catálogo de productos de forma inmediata y con posibilidad de edición sin afectar la fuente de información.

- Con una interfase de consulta rápida.

- Total control de los datos con posibilidades de cambios o correcciones.

- Número ilimitado de presupuestos.

- Número ilimitado de productos o partidas por cada presupuesto.

- Cálculo automático de importes ($).

- Impresión de Presupuestos, Clientes, Catálogo de productos.

- Restricción por niveles a capturistas para cambios de precios.

- Cantidad mínima de memoria RAM. (inicialmente se usó en XT)

- Modular. Para poder ser heredado a las siguientes etapas.

V.4. DESARROLLO DE UN MODELO PROTOTIPO

Se desarrollo un módelo para dar solución inmediata al problema al mismo tiempo que sirvió para introducir a los usuarios en el manejo de bases de datos, la teoría (MOVE) y por otro lado para alimentar el catálogo de productos básicos.

Este prototipo fue desarrollado en dBase IV usando el generador de aplicaciones.

V.5. DISEÑO DEL SISTEMA

Mientras los usuarios trabajaban con el modelo prototipo, se realizó el diseño global del sistema, utilizando como ya se ha mencionado los diagramas WARNIER-JACKSON, BLOQUES Y DE FLUJO, obteniendo incialmente los siguientes (MOVE).

- CATÁLOGO DE PRODUCTOS .
- CATÁLOGO DE CLIENTES.
- ARCHIVO DE PRESUPUESTOS.
- ARCHIVO DE PRODUCTOS DE CADA PRESUPUESTO.
- CONTROL DE USUARIOS CLAVES Y NIVELES DE ACCESO.

* PROGRAMA DE MENU (PERSIANAS) para selección de opciones.

V.6. DESARROLLO DEL SOFTWARE

El software final para esta aplicación ya comprende los módulos para FACTURAS, PEDIDOS, REMISIONES, FATURAS Y ORDENES DE COMPRA, esta escrito en lenguaje xBase y compilado con Clipper 5.0.

Si se desea consultar el código fuente recurra al **APENDICE B** donde se muestran algúnos de los programas de esta aplicación.

V.7. PRUEBA DEL SISTEMA

Al terminar de desarrollar el sistema se ralizan algunas pruebas sencillas como :

- Recorrer cada una de las opciones o acciones que debe realizar el sistema asegurándonos que realiza lo esperado.
- Presionar teclas que el progrma no espera
- Intentar sobrepasar límites de las bases de datos.

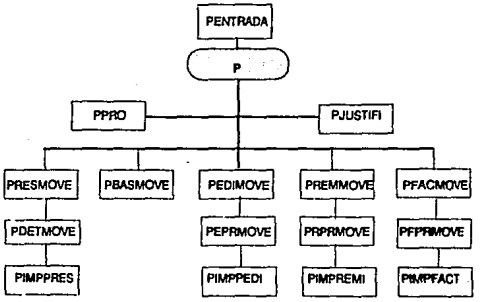
Cuando el sistema estuvo en condiciones de ser probado y operado por los usuarios finales, substituimos las bases de datos por las ya capturadas en el prototipo, haciendo varios respaldos en disco duro y dikettes, aunque estos no se necesitaron debido a que el sistema no tuvo problemas en la ejecución.

V.8. PUESTA EN MARCHA

Una vez terminado el sistema, reemplazó al modelo prototipo, logrando una gran aceptación por los usuarios, y a lo largo de su uso se han solicitado innumerables mejoras tanto para agilizar el proceso administrativo como para la teoría MOVE.

V.9. DESCRIPCIÓN GENERAL DE LOS PROGRAMAS PRINCIPALES

El siguente esquema presenta una vista general del uso y colocación de algunos de los proramas del sistema y a continuación se da una descripción de lo que cada uno de estos programas realiza dentro del sistema.



PENTRADA.PRG: Aunque no es el programa de carga inicial, es llamado por el principal P.PRG y es lo primero que veremos en pantalla al ejecutar P.EXE. Este programa contiene la declaración de variables públicas para el manejo de variables generales a las bases de datos, como apuntadores, claves de relación y colores generales del sistema. Asimismo es el programa PASSWORD para definir el nivel de acceso del usuario al sistema.

P.PRG : Es el módulo principal del sistema ya que constituye el menú de persianas para dar acceso a todos los procesos del sistema. Dentro de éste se define la liga con el programa PPRO.PRG, y la línea #INCLUDE 'DBFNDX.CH' para hacer referencia a la librería de control de archivos (.NDX).

PPRO.PRG : Contiene una colección de procedimientos y funciones generales utilizadas por y desde otros módulos del sistema.

PRESMOVE.PRG : Módulo (MOVE) de control de la base de datos PRESBASE.DBF que sólo contiene datos generales del presupuesto y no guarda los registros correspondientes a cada partida del mismo.

PDETMOVE.PRG : Módulo (MOVE) para tener el control y apuntadores relativos entre los registros de la base PDETALLE.DBF (partidas del presupuesto) y un solo registro de la base de presupuestos.

PIMPPRES.PRG : Programa de impresión del presupuesto.

PBASMOVE.PRG : Módulo (MOVE) de control de la base de datos PBASICOS.DBF que contiene el catálogo básico de productos con la descripción en campo MEMO de cada uno.

PEDIMOVE.PRG : Módulo (MOVE) de control de la base de datos PEDIDOSE.DBF que sólo contiene datos generales del pedido y no guarda los registros correspondientes a cada partida del mismo.

PEPRMOVE.PRG : Módulo (MOVE) para tener el control y apuntadores relativos entre los registros la base PEDIPROD.DBF (partidas del pedido) y un solo registro de la base de pedidos.

PIMPPEDI.PRG : Programa de impresión del pedido.

PREMMOVE.PRG : Módulo (MOVE) de control de la base de datos PREMISIO.DBF que sólo contiene datos generales de la remisión y no guarda los registros correspondientes a cada partida de la misma.

PRPRMOVE.PRG : Módulo (MOVE) para tener el control y apuntadores relativos entre los registros de la base PREMPROD.DBF (partidas de la remisión) y un sólo registro de la base de remisiones.

PIMPREMI.PRG : Programa de impresión de la remisión.

PFACMOVE.PRG : Módulo (MOVE) de control de la base de datos PFACTURA.DBF que sólo contiene datos generales de la factura y no guarda los registros correspondientes a cada partida de la misma.

PFPRMOVE.PRG : Módulo (MOVE) para tener el control y apuntadores relativos entre los registros de la base PFACPROD.DBF (partidas de la factura) y un sólo registro de la base de facturas.

PIMPFACT.PRG : Programa de impresión de la factura.

PJUSTIFI.PRG : Rutina de impresión justificada para campos MEMO con ajuste al ancho de columna de impresión. Realizada por Roberto Barrón Cortés.

V.10. CONCLUSIONES DEL CAPÍTULO

Hasta la fecha éste sistema comó algunos más de esta metodología, ha realizado todas las tareas para las cuales fue diseñado y los usuarios han obtenido los resultados esperados. La compañia usuario del sistema obtiene ahora presupuestos inmediatos para entrar en concursos con la seguridad de que presentan lo que realmente entregan a los precios correctos, y como consecuencia (de la modularidad del sistema) al ganar un concurso, la elaboración de pedidos, remisiones y facturas es para ellos labor de minutos.

CONCLUSIONES

La metododlogía MOVE nació como fruto de una recurrente necesidad de mejorar el manejo de información y facilitarle al usuario las tareas cotidianas, y por consecuencia análoga, lo que hagamos para dar bienestar generara nuevos satisfactores.

Hasta el momento todos los sistemas en los que he involucrado la teoría MOVE han cumplido con las especificaciones da cada una al mismo tiempo que han sido aceptadas con gusto por los usuarios, cabe mencionar que algunos de ellos no habían usado antes un sistema de cómputo.

Un ejemplo de lo anterior : Desarrollé un sistema (T), las entrevistas y requerimientos se hicieron con la persona que en ese momento tenía la responsabilidad, en las pruebas finales el sr. "X" participó y dio su visto bueno. Pero el día de la puesta en marcha realizaron cambios en la empresa y dejaron como responsable del área al sr. "Z" con una secretaria"W" que no conocía de sistemas, la consigna era entregar la relación de productos, proveedores y costos de aproximadamente 600 pedidos para ese mismo día. Entregué y capacité a la srita. "W" entre las 9 y 10 am. como la incertidumbre era mayúscula regresé a las 5 pm y encontre a la srita. "W" trabajando en otras cosas puesto que ya había terminado.

Esto sucedio en un grupo industrial que elabora alimentos para pasajeros de AEROLÍNEAS.

El TIEMPO que logramaos optimizar con MOVE en ocasiones no lo han querido aceptar incluso algunos expertos. En una institución BANCARIA estaban desarrollando un sistema que tenían sin entregar desde 1 año 6 meses atrás. Implantamos los MOVE al análisis y diseño que ya habían realizado y en menos de 1 mes la puesta en marcha de éste se realizo.

El CÓDIGO que actualmente se usa en la versión 6 del MOVE optimiza y substitye en promedio a 4 o 5 programas del mismo tamaño, con otra ventaja intrínseca, al momento de hacer correcciones se hacen en un sólo programa.

Un aspecto importante a recordar es que la metodología MOVE ha pasado por muchos cambios desde la versión 1 hasta la 6, lo que nos dice que ademas de ser útil se ha mejorado. Lo mejor será que aún se pueda mejorar.

El punto terminal de ésta propuesta está en el balance COSTO - BENEFICIO que podemos obtener utilizando (MOVEs) en el desarrollo de sistemas. Debido a que utilizaremos poco tiempo en el desarrollo disminuyendo con esto el COSTO y TIEMPO de programación, y como resultado de hacer sistemas más eficientes en una menor cantidad de tiempo, el BENEFICIO estará en que la cantidad de sistemas en que podemos intervenir será aun mayor, y aún así nos dejará tiempo para investigación de mejores métodos.

Por otro lado desde el punto de vista del cliente el COSTO-BENEFICIO es igualmente positivo ya que a menor número de horas de desarrollo, será menor el costo de mano de obra invertido en el desarrollo de sistemas, y por lo mismo, la solución a sus requerimientos será obtenida en en tiempo mucho más razonable.

Por último quisiera que este trabajo usado actualmente por un grupo de desarrolladores de la generación ICOM-84 (UNAM-ENEP- ARAGON) sea puesto a disposición pública para facilitar cada vez más el trabajo cotidiano de las bases de datos y la expectativa de que se convierta en un manejo estandar.

GLOSARIO

ALEATORIO : Método de acceso directo a la información por coordenadas, ya sean renglones y columnas o pistas y sectores.

APLICACION : Nombre genérico para programas o sistemas de un uso específico.

ARCHIVO : Almacén de información.

ARCHIVO ALEATORIO : Localidad en medio magnetico que permite lectura y escritura directamente sobre un registro.

ARCHIVO BINARIO : Archivo con información legible en código de máquina (1 / 0).

ARCHIVO SECUENCIAL : Localidad en la que para llegar al registro (n) se debe recorrer desde el primero.

BASE DE DATOS : Archivo aleatorio con estructura de campos y registros. Serie de información a fin acerca de un tema, organizada de manera práctica, que suministra una base o fundamento para procedimientos como la recuperación de información, elaboración de conclusiones y la toma de decisiones.

BAT : Archivo de procesamiento por lotes. Donde se pueden ejecutar en secuencia diferentes programas, comandos o aplicaciones.

BINARIO : Lenguaje usado en sistemas electrónicos para interpretar pulsos encendido y apagado (1 / 0).

BUFFER : Memoria intermedia entre una orden y su ejecución. Area de almacenamiento temporal de información.

CACHE DE DISCO : Parte de la memoria utilizada exclusivamente para guardar información leída de un disco.

CAMPO : Conjunto de datos de una misma especie.

CODIGO FUENTE : Nombre que se le da al texto de instrucciones de un programa.

COMANDO : Palabra que representa una orden o acción, esta puede darse desde un menú o desde el prompt de MS-DOS.

COMPILADOR : Es un programa desarrollado con el objetivo de traducir un bloque de instrucciones en lenguaje de alto nivel a lenguaje de máquina, ejecutando todas las instrucciones hasta que revisa todo el texto.

COMPILAR : Es el proceso para traducir un programa completo al lenguaje de máquina.

CONTROL : Menú o grupo de comandos para manejar la información.

CPU : (Central Prosses Unit) Unidad central de procesamiento. Nombre que damos comúnmente a la computadora.

DATO : Unidad de información.

DBF : (Data Base File) Extensión de los archivos base de datos.

DBT : (Data Base Text) Extensión de los archivos que guardan los campos MEMO de una base de datos.

DIRECTORIO : División de la estructura organizacional de un disco donde comunmente se almacenan archivos de un uso cotidiano..

DISCO : Medio magnético de almacenamiento de información, con acceso aleatorio.

DISCO DURO : Disco que se encuentra en forma permanente en la computadora.

DISKETTE : Disco que podemos insertar en una unidad de disco para leer o guardar información.

DISPOSITIVO : Componente de hardware ej. (Monitor, Impresora, Teclado, Mouse, etc.)

DOC : Extensión usada para archivos de texto.

DOCUMENTO : Archivo de información similar a una carta, para dar explicación de algún tema.

DRIVE : Dispositivo de entrada y salida para flujo de información entre los diskettes y la computadora.

ERRORES DE LÓGICA : Son los errores que el lenguaje no detecta al momento de revisar las instrucciones del programa, y como consecuencia los más difíciles de detectar.

ERRORES DE SEMÁNTICA : Son los errores que encontramos cuando todas la palabras del programa son conocidas por el lenguaje, pero no tienen un orden o secuencia lógico.

ERRORES DE SINTAXIS : Son los errores que encontramos al escribir un programa y alguna palabra no es conocida por el lenguaje.

ERRORES LEXICOGRÁFICOS : Se refiere a que cuando escribimos el programa utilizamos caracteres no reconocidos por el lenguaje.

EXE : Extensión de un archivo o programa ejecutable.

FMT : Extensión de un archivo que guarda intrucciones de colocación de letreros y campos en la pantalla.

FORMATO : Modo en que esta organizada la información en pantalla, o la organización interna de un archivo.

FRM : Extensión de un archivo para impresión dentro de dBase.

HARDWARE : Todos los componentes que son tangibles.

INFORMACIÓN : Conjunto de datos con un orden lógico.

INTERFASE : Intervalo de acoplamiento entre dos o más etapas diferentes.

INTÉRPRETE : Es un programa desarrollado para traducir una a una las instrucciones de algún lenguaje de alto nivel al lenguaje de máquina, y de este modo realizar la operación de esta instrucción.

LENGUAJE :Conjunto sistemático de signos que permiten la comunicación y construcción de programas con los que se puede operar una computadora.

LIGAR : Es el proceso para convertir un programa compilado en un programa ejecutable.

LNK : : Extención de un archivo para ligar un programa con el precompilador de Clipper.

LPT : Siglas del puerto paralelo usado comúnmente para la impresora.

MARCADO : Situación temporal de un registro para poder ser eliminado posteriormente.

MEMORIA CACHE : Memoria para agilizar el trabajo de transporte de información.

MEMORIA EXPANDIDA : Ampliación lógica de los circuitos de memoria y que pueden accesar solo algunos programas.

MEMORIA EXTENDIDA : Memoria más allá de un megabyte, solo en computadoras 286, 386, 486, etc.

MEMORIA VIRTUAL : Sistema de memoria utilizado por algunos paquetes para simular que hay más memoria de la que realmente tenemos, usando para esto una area temporal del disco duro.

MENU : Lista de comandos o intrucciones disponibles en un recuadro de selección.

MENU BAR : Menú de selección de opciones usando teclas de cursor izquierda y derecha.

MENU PERSIANAS : Menú de selección de opciones por recuadros de aparición súbita, usando teclas de cursor arriba, abajo, izquierda y derecha.

MENU POP-UP : Menú de selección de opciones usando teclas de cursor arriba y abajo.

MENU PULL-DOWN : Menú de selección de opciones usando teclas de cursor arriba, abajo, izquierda y derecha.

MIDI : (Musical Instrument Digital Interface) Protocolo estandar para comunicar instrumentos musicales con la comptadora.

MODEM : (MOdulador DEModulador) Dispositivo que permite transmitir o recibir información en una computadora por vía telefónica.

MODULAR : Conjunto de programas interelacionados jerárquica o independientemente a un programa principal.

MOVE : Nombre de la teoría de manejo de bases de datos que involucra varios procedimientos en un solo menú de control.

MS-DOS : (Micro-Soft Disk Operating System) Nombre que recibe el sistema operativo de la plataforma PC.

MULTITAREA : Capacidad que tienen algunas computadoras para ejecutar dos o más tareas al mismo tiempo.

NDX : Extensión de un archivo de indexación (ordenamiento) en dBase.

NTX : Extensión de un archivo de indexación (ordenamiento) en Clipper.

OBJETO : Elemento con capacidad de guardar información sobre (qué es, quién lo usa, imagenes, sonido, datos dependientes, etc).

PAQUETE : Conjunto de programas para tratamiento de información en una área.

PARÁMETRO : Información o datos que se añaden a un comando para que este las use como datos de inicio.

PC : (Personal Computer) Plataforma de computadoras basadas de D.O.S.

PISTA : Sección o corte lógico circular de un disco.

PIXEL : Localidad mínima de visualización en un monitor.

PLATAFORMA : Clasificación que agrupa a las computadoras por su uso y compatibilidad.

PRG : Extensión de un archivo de programación para xBase.

PROGRAMA : Secuencia de INSTRUCCIONES escrita en algún lenguaje de computadora.

PROGRAMA OBJETO : Denominación que se le da al programa cuando esta en ejecución.

PROMPT : Punto de petición de comandos, ya sea en sistema operativo o en algún paquete.

PROTOCOLO : Conjunto de reglas que definen la forma para comunicarse con una computadora.

PUERTO : Punto de conexión entre la computadora y algún dispositivo periférico.

RAM : (Random Access Memory) Memoria de acceso aleatorio

RESIDENTE : Localidad donde se almacenan en memoria algunos programas.

REGISTRO : Conjunto de campos que forman un renglón de información.

RMK : Extensión de un archivo para crear relación entre los fuentes y los objetos.

ROM : (Read Only Memory) Memoria de sólo lectura

SCR : Extensión de un archivo para el diseño de un formato de captura en dBase.

SECTOR : Sección o corte lógico transversal (como una rebanada de pastel) de un disco.

SERIAL : Puerto de comunicación de una computadora.

SISTEMA : Programa o conjunto de programas con un fin determinado relacionados entre si.

SOFTWARE : Todo el conjunto de órdenes para que las computadoras realicen cualquier tarea.

SUBDIRECTORIO : División de la estructura organizacional de un disco y que a su vez pertenece a otro directorio o subdirectorio.

TSR : Programa que puede alojarse en memoria para actuar aún cuando se este ejecutando otro.

TXT : Extensión usada para archivos de texto.

UNIDAD DE DISCOS : Dispositivo de entrada y salida para flujo de información entre los diskettes y la computadora.

xBase : Son todos aquellos productos que están enfocados al manejo de bases de datos.

APÉNDICE A

PROGRAMAS Y RUTINAS DE LA METODOLOGÍA DE MOVIMIENTO

CONTENIDO DEL APÉNDICE A

EJEMPLO1.PRG

```
*-------------------------------------------------*
* EJEMPLO1.PRG                                    *
* REALIZADO POR ARTURO CROTTE FRANCO.             *
* DEFINICION DEL MENU HORIZONTAL Y SUS PAD'S*
  * DEFINICION DE LOS MENUS POPUP ASOCIADO AL HORIZONTAL*
*-------------------------------------------------*
CLEAR ALL
CLOSE ALL
SET TALK OFF
SET STAT OFF
SET SCOR OFF
SET ESCAPE OFF
SET COLOR TO W+/B,N/W
DEFI WIND VENTANA FROM 5,0 TO 24,79 DOUBLE
STORE " " TO SAL
CLEAR
SET COLOR TO W/N,N/W,W
@ 2,20 CLEAR TO 8,60
@ 2,20 TO 8,60 DOUB
@ 4,25 SAY " ESPERA UN MOMENTO POR FAVOR ... "
@ 6,25 SAY " GENERANDO LOS MENUS DE CONTROL "
SET COLOR TO
DEFI MENU CENTRAL
DEFI PAD CENTRAL OF CENTRAL PROM "CENTRAL" AT 3,03
DEFI PAD PULL   OF CENTRAL PROM "POP-UP" AT 3,13
DEFI PAD BARRAS  OF CENTRAL PROM "BARRAS" AT 3,21
DEFI PAD SALIDA  OF CENTRAL PROM "SALIA"  AT 3,72

ON PAD CENTRAL OF CENTRAL ACTI POPU CENTRAL
ON PAD PULL   OF CENTRAL ACTI POPU PULL
ON PAD BARRAS  OF CENTRAL ACTI POPU BARRAS
ON PAD SALIDA  OF CENTRAL ACTI POPU SALIDA



DEFI POPU CENTRAL FROM 4,03
DEFI BAR 1 OF CENTRAL PROM " DEFINIR "
DEFI BAR 2 OF CENTRAL PROM " ACTIVAR "
DEFI BAR 3 OF CENTRAL PROM " DESACTIVAR "
ON SELE POPU CENTRAL DO CENTRAL

DEFI POPU PULL FROM 4,13
DEFI BAR 1 OF PULL  PROM " DEFINIR "
DEFI BAR 2 OF PULL  PROM " ACTIVAR "
DEFI BAR 3 OF PULL  PROM " DESACTIVAR "
ON SELE POPU PULL DO PULL

DEFI POPU BARRAS FROM 4,21
DEFI BAR 1 OF BARRAS PROM " DEFINIR "
DEFI BAR 2 OF BARRAS PROM " SELECCIONAR "
ON SELE POPU BARRAS DO BARRAS

DEFI POPU SALIDA FROM 4,66
 DEFI BAR 1 OF SALIDA PROM " FINALIZAR " MESS "TERMINA LA SECION DE TRABAJO Y REGRESA AL SISTEMA OPERATIVO"
DEFI BAR 2 OF SALIDA PROM REPL("D",11) SKIP
DEFI BAR 3 OF SALIDA PROM " PROGRAMA "
ON SELE POPU SALIDA  DO SALIDA

DO WHIL .T.
  CLEAR
  SET COLOR TO
  @ 2,1 TO 4,79 DOUB COLOR N/W
  DO TEXTO
  ACTI MENU CENTRAL
  IF SAL = "SI"
    EXIT
  ENDI
  DEAC MENU
ENDD
RELE WIND VENTANA
CLOSE ALL
SET STAT ON
SET TALK ON
SET COLOR TO
SET ESCAPE ON
CLEAR


*-------------------------------------------------*
*     PROCEDIMIENTOS ASOCIADOS A LA SELECCION DE LOS MENUS *
*-------------------------------------------------*


PROC TEXTO
*
* LOGOTIPO DEL SISTEMA
*
RETURN
*-------------------------*
PROC CENTRAL
ACTI WIND VENTANA
DO CASE
  CASE BAR() = 1
    TEXT
    EJEMPLO  EN LA DEFINICION DE UN MENU  CENTRAL Y
    LA DECLARACION DE ITEMS (PAD) QUE SE INCLUYEN
    DENTRO DEL MENU ASI COMO LA POSICION DONDE IRAN
    APARECIENDO CADA UNO DE ELLOS Y EL MENU QUE SE

    ACTIVA AL TOCAR ESTE ITEM.

    DEFI MENU CENTRAL
     DEFI PAD CENTRAL  OF CENTRAL PROM "CENTRAL" AT
3,03
    DEFI PAD PULL   OF CENTRAL PROM "POP-UP" AT 3,13
     DEFI PAD BARRAS  OF CENTRAL PROM "BARRAS" AT
3,21
    DEFI PAD SALIDA  OF CENTRAL PROM "SALIA"  AT 3,72

    ON PAD CENTRAL OF CENTRAL ACTI POPU CENTRAL
    ON PAD PULL   OF CENTRAL ACTI POPU PULL
    ON PAD BARRAS  OF CENTRAL ACTI POPU BARRAS
    ON PAD SALIDA  OF CENTRAL ACTI POPU SALIDA
    ENDT
    READ
  CASE BAR() = 2
    TEXT

    ACTI MENU CENTRAL


    ENDT
    READ
```

```
    CASE BAR() = 3
      TEXT



      DEAC MENU

      ENDT
      READ
  ENDC
  DEAC WIND VENTANA
  RETU
  *-------------------------*
  PROC PULL
  ACTI WIND VENTANA
  DO CASE
    CASE BAR() = 1
      TEXT

      DEFI POPU CENTRAL FROM 4,03
      DEFI POPU PULL   FROM 4,13
      DEFI POPU BARRAS  FROM 4,21
      DEFI POPU SALIDA  FROM 4,6
        DEFI BAR  1 OF SALIDA PROM ' FINALIZAR '  MESS
'TERMINA LA ...'
      DEFI BAR 2 OF SALIDA PROM REPL('D',11)  SKIP
      DEFI BAR  3 OF SALIDA PROM ' PROGRAMA '
      ON SELE POPU SALIDA  DO SALIDA

      DEFI POPUP  [NOMBRE]
      ENDT
      READ
    CASE BAR() = 2
      TEXT

      ON PAD CENTRAL  OF CENTRAL ACTI POPU CENTRAL
      ON PAD PULL   OF CENTRAL ACTI POPU PULL
      ON PAD BARRAS  OF CENTRAL ACTI POPU BARRAS
      ON PAD SALIDA  OF CENTRAL ACTI POPU SALIDA


      ACTI POPUP  [NOMBRE]
      ENDT
      READ
    CASE BAR() = 3
      TEXT


      DEAC POPUP PULL

      ENDT
      READ
  ENDC
  DEAC WIND VENTANA
  RETU
  *-------------------------*
  PROC BARRAS
  ACTI WIND VENTANA
  DO CASE
    CASE BAR() = 1
      TEXT

      DEFI BAR  1 OF BARRAS  PROM ' DEFINIR '
      DEFI BAR  2 OF BARRAS  PROM ' SELECCIONAR '

      DEFI BAR (N) OF (NOMBRE DEL POPUP) PROMP 'TITULO
DEL ITEM'

      ENDT
      READ
    CASE BAR() = 2
      TEXT
       ON SELE POPU BARRAS DO BARRAS
       ON SELE POPU (NOMBRE DEL POPUP) DO (RUTINA DE
SELECCION)

      ENDT
      READ
  ENDC
  DEAC WIND VENTANA
  RETU
  *-------------------------*
  *-------------------------*
  PROC SALIDA
  DO CASE
    CASE BAR() = 1
      SAL = 'SI'
      DEFI WIND SALID FROM 11,20 TO  15,60 DOUB
      ACTI WIND SALID
        @ 1,0 SAY 'REALMENTE DESEA TERMINAR LA SESION?'
GET SAL PICT '@M SI,NO'
      READ
      DEAC WIND SALID
      RELE WIND SALID
      IF SAL = 'SI'
        CLOSE ALL
        QUIT
      ENDIF
      RETU
    CASE BAR() = 2
      *
    CASE BAR() = 3
      SAL = 'SI'
      DEAC MENU
      CLOSE ALL
      RETU
  ENDCASE
  *-------------------------------------------------*
  *-------------------------------------------------*
  *            FIN DE PROCEDIMIENTOS ASOCIADOS A LOS
MENUS
  *-------------------------------------------------*
```

TECLAS.PRG

```
*************************************************
* PROGRAMA.....: TECLAS.PRG
* NOTAS.....: INTERCEPCION DE TECLAS Y PRESENTACION EN ASCII
* BASE DE DATOS.....:
* INDEX .....:
* SELECT .....:
* FECHA .....:
* VERSION .....: 1.0
* REALIZADO POR : ARTURO CROTTE FRANCO
*************************************************

 SET ESCA OFF     && DESACTIVA ESCAPE PARA OBTENERE VALORES CORRECTOS
 SET TALK OFF     && DESACTIVA MENSAJES DE ASIGNACION
 SET STAT OFF     && DESACTIVA BARRA DE STATUS PARA DBASE
 CLOSE ALL        && CIERRA BASES FORMATOS E INDICES
 CLEAR ALL        && LIMPIA VARIABLES DE MEMORIA
 CLEAR            && LIMPIA PANTALLA
 T = 0            && INICIA VARIABLE T EN CERO
 @ 21,30 SAY "PRESIONA ESC PARA TERMINAR"
 @ 10,14 SAY "PRESIONA CUALQUIER TECLA."
 DO WHILE T # 27  && INICIA CICLO HASTA TECLA ESC
  T = 0           && RESIGNA CERO A T
  DO WHILE T = 0  && INICIA CIPLO PARA T IGUAL A CERO
   T = INKEY()    && ASIGNA VALOR ASCII DE LA TECLA A LA VARIABLE
  ENDDO           && CIERRE DEL CICLO T = 0
  @ 10,40 SAY TRIM(STR(T)) + " = " + CHR(T)
 ENDDO            && CIERRE DEL CICLO PARA T IGUAL A ESC
 CLEAR
 CLEAR ALL
 SET ESCA ON      && REASIGNACION DE VALORES DEFOULT
 SET TALK ON
 SET STAT ON
```

CURSOR.PRG

```
*************************************************
* PROGRAMA.....: CURSOR.PRG
* NOTAS.....: CONTROL Y MOVIMIENTO DEL CURSOR
* FECHA .....:
* VERSION .....: 1.0
* REALIZADO POR     : ARTURO CROTTE FRANCO
*************************************************
 SET ESCA OFF    && DESACTIVA ESCAPE PARA OBTENERE VALORES CORRECTOS
 SET TALK OFF    && DESACTIVA MENSAJES DE ASIGNACION
 SET STAT OFF    && DESACTIVA BARRA DE STATUS PARA DBASE
 CLOSE ALL       && CIERRA BASES FORMATOS E INDICES
 CLEAR ALL       && LIMPIA VARIABLES DE MEMORIA
 CLEAR           && LIMPIA PANTALLA
 STORE 0 TO T,C,R && INICIA VARIABLES T,R,C EN CERO
 *               C ES PARA COLUMNA
 *               R ES PARA RENGLON
 C = 40          && VALORES PARA INICIO EN EL CENTRO DE LA PANTALLA
 R = 12
 DO WHILE T # 27  && INICIA CICLO HASTA TECLA ESC
   T = 0         && RESIGNA CERO A T
   @ R,C SAY T   && PINTA CARACTER 219 EN RENGLON Y COLUMNA
   DO WHILE T = 0  && INICIA CIPLO PARA T IGUAL A CERO
     T = INKEY()   && ASIGNA VALOR ASCII DE LA TECLA A LA VARIABLE
   ENDDO         && CIERRE DEL CICLO T = 0
   * @ R,C SAY T  && BORRA CARACTER 219 EN RENGLON Y COLUMNA
   DO CASE       && SELECCION DE TECLA PRESIONADA
     CASE T = 1  && TECLA HOME  1 = DBJ  29 = DBH
       R = 0     && EXTREMO SUPERIOR IZQUIERDO
       C = 0
     CASE T = 6  && TECLA END   6 = DBJ  2 = DBH
       C = 79    && EXTREMO INFERIOR DERECHO
       R = 24
     CASE T = 18 && TECLA PGUP
       R = 0     && INICIO DE COLUMNA
     CASE T = 3  && TECLA PGDN
       R = 24    && FIN DE COLUMNA
     CASE T = 24 && TECLA FLECHA ABAJO
       IF R = 24 && SELECCION PARA HACER CIRCULAR EL RENGLON
         R = 0
       ELSE
         R = R + 1
       ENDI
     CASE T = 5  && TECLA FLECHA ARRIBA
       IF R = 0  && SELECCION PARA HACER CIRCULAR EL RENGLON
         R = 24
       ELSE
         R = R - 1
       ENDI
     CASE T = 4  && TECLA FLECHA DERECHA
       IF C = 79 && SELECCION PARA HACER CIRCULAR LA COLUMNA
         C = 0
       ELSE
         C = C + 1
       ENDI
     CASE T = 19 && TECLA FLECHA IZQUIERDA
       IF C = 0  && SELECCION PARA HACER CIRCULAR LA COLUMNA
         C = 79
       ELSE
         C = C - 1
       ENDI
     CASE T = 27 && TECLA ESC
       CLEAR     && LIMPIAR PANTALLA ANTES DE SALIR

     OTHE
           * NO HAY ACCION
   ENDC          && FIN DE SELECCION DE TECLAS
 ENDD            && CIERRE DEL CICLO PARA T IGUAL A ESC
 CLEAR
 CLEAR ALL
 SET ESCA ON     && REASIGNACION DE VALORES DEFOULT
 SET TALK ON
 SET STAT ON
```

ESTADO.PRG

```
*************************************************************
* PROGRAMA.........: ESTADO.PRG
* NOTAS............: POSICION, LIMITES Y MOVIMIENTO EN UN DBF
*  BASE DE DATOS...: ABIERTA DES FUERA DE ESTE PROGRAMA
* INDEX............:
* SELECT...........: 1
* FECHA............:
* VERSION..........: 1.0
* REALIZADO POR    : ARTURO CROTTE FRANCO
*************************************************************
SET ESCA OFF
SET DELE OFF
SET TALK OFF
SET STAT OFF
STORE 0 TO REA,REI,REF,T
GO BOTT
REF = RECNO()
GO TOP
REI = RECNO()
REA = RECNO()
********** LOOP - PRINCIPAL **********
DO WHIL T#27
  IF RECC()#0
    GO REA
  ENDI
  SET COLOR TO N/W
  @ 23,00 CLEAR TO 24,79
  DO CASE
    CASE RECC() = 1
      @ 24,30 SAY ' 1 SOLO REGISTRO    '
    CASE RECC() = 0
      @ 24,20 SAY ' ARCHIVO SIN DATOS    '
    CASE REA = REF
      @ 24,30 SAY ' FIN DE ARCHIVO    '
    CASE REA = REI
      @ 24,30 SAY ' PRINCIPIO DE ARCHIVO '
  ENDC
  @ 24,5 SAY 'REG : ' + LTRIM(STR(RECNO())) + '/' + LTRIM(STR(RECC()))
  IF DELETED()
    SET COLOR TO N*/W
    @ 24,20 SAY ' MARCADO '
  ENDI
  SET COLOR TO N/W
  @ 23,00 SAY ' USAR : ' + CHR(25) + CHR(24) + ' HOME PGDN PGUP  ESC'
  T = 0
  DO WHILE T = 0
    T = INKEY()
  ENDD
  @ 23,00 CLEAR TO 24,79
  DO CASE
    CASE T = 1   .AA.   TECLA   HOME     1 = DBS    26 = DB4
      GO TOP
      REA = RECNO()
    CASE T = 6   .AA.   TECLA   END      6 = DBS    2 = DB4
      GO BOTTOM
      REA = RECNO()
    CASE T = 18  .AA.   TECLA   PGUP
      S = 0
      DO WHILE .NOT. BOF()
        SKIP -1
        REA = RECNO()
        S = S + 1
        IF S = 10
          EXIT
        ENDI
      ENDDO
    CASE T = 3   .AA.   TECLA   PGDN
      S = 0
      DO WHILE .NOT. EOF()
        SKIP
        REA = RECNO()
        S = S + 1
        IF S = 10
          EXIT
        ENDI
      ENDD
    CASE T = 24  .AA.   TECLA   FLECHA ABAJO
      IF REA = REF
        ?? CHR(7)
      ELSE
        SKIP 1
        REA = RECNO()
      ENDI
    CASE T = 5   .AA.   TECLA   FLECHA ARRIBA
      IF REA = REI
        ?? CHR(7)
      ELSE
        SKIP -1
        REA = RECNO()
      ENDI
    CASE T = 27  .AA.   TECLA   ESC
      CLEAR
    OTHERWISE
      ?? CHR(7)
  ENDC
ENDD
SET COLOR TO
CLEAR
CLEAR ALL
SET STAT ON
SET TALK ON
********** FIN DEL LOOP - PRINCIPAL **********
```

BUSCAR.PRG

```
*************************************************************
 * PROGRAMA.....: BUSCAR.PRG
 * NOTAS.....: IDENTIFICAR EL CAMPO INDEX Y BUSCAR POR EL
 * BASE DE DATOS.....: PRUEBA (CAMPOS / FECHA / NOMBRE / CIUDAD)
 * INDEX .....: SELECCION EXTERNA
 * SELECT .....: 1
 * FECHA .....:
 * VERSION .....: 1.0
 * REALIZADO POR     : ARTURO CROTTE FRANCO
 *************************************************************

 #INCLUDE "DBFNDX.CH"

  SET WRAP ON                    && ACTIVA MENU EN FORMA CIRCULAR
  SET DATE ITALIAN               && CAMBIO DE FORMATO DE FECHA
  DO WHIL .T.
    SET COLOR TO W+/B,GR+/N,W
    CLEAR
    @ 02,20 SAY " SELECCIONA EL CAMPO A BUSCAR "
    @ 04,30 PROM " FECHA "
    @ 05,30 PROM " NOMBRE "
    @ 06,30 PROM " CIUDAD "
    MENU TO OP
    DO CASE
      CASE OP = 0
        EXIT
      CASE OP = 1
        USE PRUEBA INDEX PRUEFECH,PRUECIUD,PRUENOMB VIA "DBFNDX"
      CASE OP = 2
        USE PRUEBA INDEX PRUENOMB,PRUEFECH,PRUECIUD VIA "DBFNDX"
      CASE OP = 3
        USE PRUEBA INDEX PRUECIUD,PRUEFECH,PRUENOMB VIA "DBFNDX"
    ENDC
    REA = RECN()
        ********** BUSQUEDA SEEK FIND LOCATE **********
    @ 23,00 CLEAR
    @ 24,5 SAY "REG : " + LTRIM(STR(RECN())) + "/" + LTRIM(STR(RECC()))
    @ 23,30 SAY "BUSCAR POR " + INDEXKEY()
    DO CASE
      CASE "FECHA"$INDEXKEY()
        MFI = FECHA
        @ 04,45 GET MFI
        READ
        IF LASTKEY() = 27
          LOOP
        ENDI
        SEEK MFI
        IF FOUND()
          REA = RECN()
        ELSE
          GO TOP
          * BUSQUEDA EXAUSTIVA A MES Y A%O SIN IMPORTAR DIA
          LOCA FOR MONT(MFI) = MONT(FECHA) .AND. YEAR(MFI) = YEAR(FECHA)
          IF FOUND()
            REA = RECN()
          ENDI
        ENDI
      CASE "NOMBRE"$INDEXKEY()
        MNOMBRE = SPACE(LEN(NOMBRE))
        @ 05,45 GET MNOMBRE PICTURE "@!" VALID .NOT. EMPTY(MNOMBRE)
        READ
        MNOMBRE = ALLTRIM(MNOMBRE)
        IF LASTKEY() = 27 .OR. EMPTY(MNOMBRE)
          LOOP
        ENDI
        FIND &MNOMBRE
        IF FOUND()
          REA = RECN()
        ELSE
          GO TOP
          LOCA FOR MNOMBRE=&NOMBRE
          IF FOUND()
            REA = RECN()
          ENDI
        ENDI
      CASE "CIUDAD"$INDEXKEY()
        MCIUDAD = SPACE(LEN(CIUDAD))
        @ 05,45 GET MCIUDAD PICTURE "@!" VALID .NOT. EMPTY(MCIUDAD)
        READ
        MCIUDAD = ALLTRIM(MCIUDAD)
        IF LASTKEY() = 27 .OR. EMPTY(MCIUDAD)
          LOOP
        ENDI
        FIND &MCIUDAD
        IF FOUND()
          REA = RECN()
        ENDI
      ENDC
      GO REA
      @ 24,50 SAY "REG : " + LTRIM(STR(RECN())) + "/" + LTRIM(STR(RECC()))
      @ 15,10 SAY "FECHA    " + DTOC(FECHA)
      @ 16,10 SAY "NOMBRE   " + NOMBRE
      @ 17,10 SAY "DIRECCION " + DIRECCION
      @ 18,10 SAY "COLONIA  " + COLONIA
      @ 19,10 SAY "CIUDAD   " + CIUDAD
      @ 20,10 SAY "C.P.     " + CP
      INKEY(0)
    ENDC
    CLOSE ALL
    SET COLOR TO
    CLEAR
```

VALIDAR.PRG

```
*************************************************
* PROGRAMA.....: VALICAR.PRG
  * NOTAS.....: RUTINA DE VALIDACION PARA NO DUPLICAR
* BASE DE DATOS.....:
* INDEX .....:
* SELECT .....:
* FECHA .....:
* VERSION .....: 1.0
* REALIZADO POR     : ARTURO CROTTE FRANCO
*************************************************
* VALIDA
  MBUSCA = CAMPO          && ASIGNAR LA VARIABLE DE CAMPO A MEMORIA
 @ Y,X GET MBUSCA
 READ
 IF LASTKEY() = 27
   LOOP
 ENDI
 *     SELECCIONAR EL METODO DE BUSQUEDA
 *  LOCA FOR MBUSCA=&CAMPO
 *  SEEK MBUSCA
 *  FIND &MBUSCA
 IF FOUND()
   ?? CHR(7)
   SET COLOR TO &C_RESALTA
   @ 23,00 CLEAR
   @ 23,10 SAY " !!! ATENCION !!! IDENTIFICADOR REPETIDO."
   @ 24,10 SAY " PRESIONA CUAQUIER TECLA PARA CONTINUAR..."
   INKEY(0)
   REA = RECNO()
   LOOP
 ENDI
 APPE BLANK
 *        REPL CAMPO WITH MBUSCA
 SET COLOR TO &C_CAJA
 @ 23,00
   @ 23,10 SAY " AGREGANDO UN REGISTRO AL ARCHIVO. (ALTA)"
 SET COLOR TO &C_ENTRA
 REA = RECNO()
 ********** GET CAMPOS DE CAPTURA **********
```

TRAERDAT.PRG

```
********************************************************
* PROGRAMA.........: TRAERDAT.PRG
* NOTAS..........: EJEMPLO PARA TRAER DATOS DE UNA
BASE A OTRA
*               : USANDO TABLAS Y TECLAS DE FUNCION
* BASE DE DATOS.....: PRESBASE      PCLIENTE
* INDEX ..........: PRESBASE.NOX   PCLIENTE.NOX
* SELECT ..........: 1          1
* FECHA ..........:
* VERSION ..........: 1.0
* REALIZADO POR      : ARTURO CROTTE FRANCO
********************************************************
#INCLUDE 'DBFNDX.CH'

C_CAJA   = "W/B ,BG/N ,N,,W/B "
C_TEXT   = "N + /W,BG + /N,N,,W + /B"
C_ENTRA  = "N/BG,BG + /N,N,,N/BG"
C_VENTANA = "BG/N,B/N ,N,,B + /N"
C_RESALTA = "W + /B,BG + /N,N,,W + /B"

********************************
*** TABLA DE CLIENTES         ***
********************************
SELE 1
USE PCLIENTE INDEX PCLIENTE VIA 'DBFNDX'
TOTAL = RECC()
RELEASE MCLIENTES
DECLARE MCLIENTES[TOTAL]
MC = 1
GO TOP
DO WHILE .NOT.EOF()
  MCLIENTES[MC] = NOMBRE
  SKIP
  MC = MC + 1
ENDDO
USE

SELE 1
USE PRESBASE INDEX PRESBASE VIA 'DBFNDX'
DO WHIL .T.
   ************** FORMACION DE PANTALLA DE
VISUALIZACION ****************
  SET COLOR TO N + /W,BG + /N,B + ,,N/BG
  CLEAR
  SET COLOR TO B + /B
  @ 0, 0 SAY REPLICATE('\',80)
  SET COLOR TO W + /B
  @ 0,33 SAY ' PRESBASE.DBF '
  SET COLOR TO N + /W
  @ 1,49 TO 7,78 DOUBLE
  @ 2,13 SAY 'DISE%O PROFESIONAL DE EQUIPOS'
  @ 2,54 SAY 'P R E S U P U E S T O'
  @ 3,16 SAY 'Y COCINAS, S.A. DE C.V.'
  @ 4,51 SAY 'REF'
  @ 5,12 SAY 'N#MERO DE PRESUPUESTO :'
  @ 6,12 SAY 'TOTAL DE PRESUPUESTOS :'
  @ 5,51 SAY 'FECHA'
  @ 6,51 SAY 'AGENTE'
  @ 8,2 TO 20,78 DOUBLE
  @ 9,5 SAY 'NOMBRE.....'
  @ 10,5 SAY 'ATENCION...'
  @ 11,5 SAY 'DIRECCION..'
  @ 12,5 SAY 'TELEFONOS..'
  @ 13,5 SAY 'PLAZO DE'
  @ 14,5 SAY 'ENTREGA....'
  @ 15,5 SAY 'FORMA DE...'
  @ 16,5 SAY 'P A G O....'
  @ 18,5 SAY 'OBSERVACIONES:'

  MNOMBRE = NOMBRE
  SET KEY -8 TO CON_CLIE
  SET COLOR TO &C_RESALTA
  @ 23,00
     @ 23,10 SAY ' EDITANTO DATOS DEL REGISTRO
F9 = CLIENTES'
  SET COLOR TO &C_ENTRA
  @ 5,36 SAY RECNO() PICTURE '999,999'
  @ 5,62 GET FECHA
  @ 6,62 GET AGENTE PICTURE '@S15'
  @ 9,16 GET MNOMBRE PICT '@!'
  READ
  SET KEY -8 TO
  IF LASTKEY() = 27
    EXIT
  ENDI
  SET COLOR TO &C_CAJA
  @ 23,00
  @ 23,10 SAY ' EDITANDO DATOS DEL REGISTRO '
  SET COLOR TO &C_ENTRA
  @ 10,16 GET ATENCION  PICTURE '@!'
  @ 11,16 GET DIRECCION PICTURE '@!'
        @ 12,16 GET TELEFONOS PICTURE
'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
  @ 14,16 GET PLAZO  PICTURE '@!'
  @ 15,16 GET FORMA1 PICTURE '@!'
  @ 16,16 GET FORMA2 PICTURE '@!'
  @ 17,16 GET FORMA3 PICTURE '@!'
  @ 19, 7 GET NOTAS  PICTURE '@!'
  REPL NOMBRE WITH MNOMBRE
  READ
ENDD
SET COLOR TO
CLEAR
CLOSE ALL

**************************************************************
**************************************************************
**************************************************************
*          CONSULTA DE CLIENTES
*
PROC CON_CLIE
MSCC = SAVESCREEN(00,00,24,79)
SET COLOR TO &C_CAJA
SET KEY -8 TO
MNUM = 0
@ 23,00
@ 23,00 SAY ' FLECHAS = SELECCIONAR  ENTER = ACEPTAR
ESC = ABANDONAR  '
@ 10,05 CLEAR TO 21,75
@ 10,05 TO 21,75 DOUBLE
@ 10,30 SAY ' C L I E N T E S '
MNUM = ACHOICE(11,13,20,73,MCLIENTES)
```

```
@ 23,00
RESTSCREEN(00,00,24,79,MSCC)
RELEASE MSCC
IF MNUM = 1 .AND. M->NUM
  MNOMBRE = MCLIENTES[MNUM]
ENDIF
SET KEY -8 TO CON_CUE
SET COLOR TO &C_ENTRA
RETU
*****
*****
*****
```

MOVE01.PRG

```
*************************************************************
* PROGRAMA.....: MOVE.PRG
* NOTAS.....: MOVIMIENTO ALTAS Y BAJAS CAMBIOS
*
* BASE DE DATOS.....:
* INDEX .....:
* SELECT .....:
* FECHA .....:
* VERSION .....: 2.1
* REALIZADO POR      : ARTURO CROTTE FRANCO
*************************************************************
SET PROCEDURE TO MOVE
SET ESCAPE OFF
SET CARRY ON
SET EXACT OFF
SET DELE OFF
SET TALK OFF
SET STAT OFF
STORE 0 TO REA,REI,REF,T
IF RECC() = 0
  DO TEXTO
  APPE BLANK
  DO MUESTRA
  T = ASC("C")
  DO CAPTA
ENDIF
GO BOTT
REF = RECNO()
GO TOP
REI = RECNO()
REA = RECNO()
DO TEXTO
********************* LOOP - PRINCIPAL *********************
DO WHIL .T.
  GO REA
  DO MUESTRA
  DO ESTADO
  SET COLOR TO N/W
  @ 22, 05 SAY " ALTA BUSCAR CAMBIO MARCAR SUPRIMIR " + CHR(25) + CHR(24) + " HOME ESC"
  DO TECLA
  IF T = 27
    EXIT
  ENDI
ENDD
SET COLOR TO
CLEAR
CLEAR ALL
CLOSE ALL
SET STAT ON
SET TALK ON
RETURN
********************* FIN DEL LOOP - PRINCIPAL **********
********PROCEDIMIENTOS ASOCIADOS AL MENU PRINCIPAL ***
PROC TEXTO
SET COLOR TO N/BG
*
*
* SECCION PARA     CUADRICA Y TEXTOS DE PANTALLA BASICA
*
*
RETU
*************************************************************
PROC MUESTRA
SET COLOR TO N/W
*
*
* SECCION @ SAY (GET) DE VARIABLES DE CAMPO
*
*
SET COLOR TO N/BG
RETURN
*************************************************************
PROC CAPTA
SET COLOR TO N/BG,N/W
IF CHR(T)$"CC"
  @ X,Y GET VARIABLE                && VARIABLE(S) DE VALIDACION
ENDI
*
*
* SECCION PARA @ GET DE VARIABLES DE CAMPO
*
*
GO TOP
REI = RECNO()
GO BOTT
REF = RECNO()
IF CHR(T)$"CC"
  REA = RECNO()
ELSE
  REA = REF
ENDI
RETU
*************************************************************
PROC ESTADO
SET COLOR TO N/W
@ 21,2 CLEAR TO 22,78
DO CASE
  CASE RECC() = 1
    @ 23,30 SAY " 1 SOLO REGISTRO    "
  CASE RECC() = 0
    @ 23,30 SAY " ARCHIVO SIN DATOS  "
  CASE REA = REF
    @ 23,30 SAY " FIN DE ARCHIVO     "
  CASE REA = REI
    @ 23,30 SAY " PRICIPIO DE ARCHIVO "
ENDC
@ 23, 5 SAY RECN() PICTURE "9999"
@ 23,10 SAY "/"
@ 23,11 SAY RECC() PICTURE "9999"
ENDC
IF DELETED()
  SET COLOR TO N*/W
  @ 23,20 SAY " MARCADO "
ENDI
SET COLOR TO N/BG
RETU
*************************************************************
PROC TECLA
  T = 0
  DO WHILE T = 0
    T = INKEY()
  ENDD
```

```
    @ 22,2 CLEAR TO 23,78
    DO CASE
       CASE T = 1   &&   TECLA   HOME        1 = CB3     26
= CB4
          GO TOP
          REA = RECNO()
       CASE T = 6   &&   TECLA   END       6 = CB3   2 = CB4
          GO BOTTOM
          REA = RECNO()
       CASE T = 18   &&   TECLA   PGUP
          S = 0
          DO WHILE .NOT. BOF()
            SKIP -1
            REA = RECNO()
            S = S + 1
            IF S = 10
              EXIT
            ENDI
          ENDDO
       CASE T = 3   &&   TECLA   PGDN
          S = 0
          DO WHILE .NOT. EOF()
            SKIP
            REA = RECNO()
            S = S + 1
            IF S = 10
              EXIT
            ENDI
          ENDDO
       CASE T = 24   &&   TECLA   FLECHA ABAJO
          IF REA = REF
            ? CHR(7)
          ELSE
            SKIP 1
            REA = RECNO()
          ENDI
       CASE T = 5   &&   TECLA   FLECHA ARRIBA
          IF REA = REI
            ? CHR(7)
          ELSE
            SKIP -1
            REA = RECNO()
          ENDI
       CASE T = 27   &&   TECLA   ESC
          T = 27
       CASE CHR(T)$'BB'
          MVARIABLE = VARIABLE
          @ X,Y GET MVARIABLE
          READ
          IF LEN(MVARIABLE) = 0 .OR. LASTKEY() = 27
            RETU
          ENDI
                SEEK MVARIABLE          &&   SI LA VARIABLE ES
NUMERICA O FECHA
                FIND &MVARIABLE         &&   SI LA VARIABLE ES
ALFANUMERICA
          IF FOUND()
            REA = RECNO()
          ENDI
       CASE CHR(T)$'AA'
          DO WHILE .T.
            @ 22,25 SAY ' A L T A '
            @ 22,40 SAY ' ESC = FIN '
            GO REF
            @ X,Y GET MVARIABLE
            READ
          IF LEN(MVARIABLE) = 0 .OR. LASTKEY() = 27
            RETU
          ENDI
                SEEK MVARIABLE          &&   SI LA VARIABLE ES
NUMERICA O FECHA
                FIND &MVARIABLE         &&   SI LA VARIABLE ES
ALFANUMERICA
          IF FOUND()
            SET COLOR TO W/R
            ?? CHR(7)
            ?? CHR(7)
            @ 23,20 SAY '   Y A  E X I S T E   '
            READ
            SET COLOR TO N/W
            @ 23,20 SAY SPACE(35)
          ELSE
            APPE BLANK
            REPLACE VARIABLE WITH MVARIABLE
            REA = RECNO()
            DO CAPTA
          ENDI
          REA = RECNO()
        ENDDO
       CASE CHR(T)$'MM'
        IF DELETED()
          RECALL
        ELSE
          DELETE
        ENDI
       CASE CHR(T)$'CC'
          DO CAPTA
       CASE CHR(T)$'SS'
          PACK
          REINDEX
          GO TOP
          REI = RECNO()
          REA = RECNO()
          GO BOTT
          REF = RECNO()
       OTHERWISE
          ?? CHR(7)
    ENDC
  RETU
  ***************************************************
  ********************* FIN DE PROCEDIMIENTOS
  ***************************************************
```

PFPRMOVE.PRG

```
*************************************************
* PROGRAMA.....: PEPRMOVE.PRG
* NOTAS.....: MOVIMIENTO ALTAS Y BAJAS CAMBIOS CONTENIDO DE
*                PRODUCTOS EN LOS FACTURAS
* BASE DE DATOS.....: PEDPROD.DBF
*          INDEX ..............:     PEDIPROD.NDX
STR(FACTURA) + NUMERO + TIPO
* SELECT .....: 3
* FECHA .....: 08-MARZO-1992
* VERSION .....: 1.0
* REALIZADO POR     : ARTURO CROTTE FRANCO
*************************************************
MP2 = SAVESCREEN(00,00,24,79)

SELE 3
STORE 0 TO REA,REI,REF,T,REL,MIMPORTE,RDA,RDF,RDI
STORE 0 TO MFACT
STORE SPACE(14) TO BUSC
MNUMERO = "  "
MTIPO = "  "
GO TOP

FIND &MREFA
IF FOUND()
  @ 23,00 CLEAR
    @ 23,00 SAY "ESPERA UN MOMENTO GENERANDO APUNTADORES..."
  REI = RECN()
  REA = RECN()
  DO WHIL FACTURA = VAL(MREFA)
    REF = RECN()
    SKIP
  ENDD
ELSE
  ?? CHR(7)
  @ 23,00 CLEAR
  @ 23,00 SAY " ERROR ...: LA FACTURA NO FUE GENERADO CON PRODUCTOS DE UN REMISION. "
    @ 24,00 SAY "   OPCION...: MARCAR Y SUPRIMIR. PRESIONA ALGUNA TECLA ..."
  INKEYTRAP(K)
  RESTSCREEN(00,00,24,79,MP2)
  RELEASE MP2
  RETU
ENDI
******* FORMACION DE PANTALLA DE VISUALIZACION *******
SET COLOR TO N + /W,BG + /N,B + ,,N/BG
CLEAR
SET COLOR TO B + /B
@ 0, 0 SAY REPLICATE("\",80)
SET COLOR TO W + /B
  MTIT = " PRODUCTOS CONTENIDOS EN LA FACTURA : " + MREFA + " "
MPTIT = 40-INT((LEN(MTIT))/2)
@ 0,MPTIT SAY MTIT
SET COLOR TO N + /W
@ 2,13 SAY "REFERENCIA..."
@ 2,28 SAY "[     ]"
@ 1,55 SAY "DESCRIPCION[          ]"
@ 2,55 SAY "TOTAL.....[          ]"
@ 3,1 TO 3,78
@ 4,1 SAY "NUMERO EN"
@ 4,10 TO 10,10
@ 4,11 SAY " -A- D E S C R I P C I O N  -A- "
@ 4,46 TO 6,46
@ 4,47 SAY "CANT."
@ 4,52 TO 18,52
@ 4,55 SAY "PRECIO"
@ 4,64 TO 18,64
@ 4,68 SAY "IMPORTE"
@ 5,1 SAY "P L A N O"
@ 5,55 SAY "/COSTO"
@ 5,66 SAY "/COSTO"
@ 6,1 TO 6,78
@ 20,2 SAY "ORDEN DE COMPRA :"
@ 21,2 SAY "STATUS :"
SET COLOR TO &C_VENTANA
@ 8,4 CLEAR TO 19,78
@ 8,4 TO 19,78
********************** LOOP - PRINCIPAL ***************
DO WHILE .T.
  IF RECC() = 0
    @ 23,00 SAY "EL ARCHIVO SE ENCUENTRA VACIO."
    @ 24,00 SAY "SE AGREGA UN REGISTRO EN BLANCO."
    INKEYTRAP(K)
    APPE BLANK
    REA = RECN()
    REF = RECN()
    REI = RECN()
  ELSE
    GO REA
  ENDI
  ********** VISUALIZACION DEL REGISTRO ACTUAL ******
  SET COLOR TO &C_ENTRA
  @ 2,27 SAY FACTURA   PICT "99999999"
  @ 1,67 SAY RECN()   PICT "999,999,999"
  @ 2,67 SAY RECC()   PICT "999,999,999"
  @ 7,2 SAY TIPO
  @ 7, 5 SAY NUMERO
  @ 7,47 SAY CANTIDAD  PICT "9,999"
  @ 7,53 SAY PRECIO   PICT "9999,999.99"
  @ 7,65 SAY IMPORTE  PICT "9999,999.99"
  @ 8,53 SAY COSTO    PICT "9999,999.99"
  @ 8,65 SAY IMPO_COST PICT "9999,999.99"
  @ 20,20 SAY ORD_COMP
  @ 21,10 SAY STATUS
  SET COLOR TO N/W
  IF TIPO = "M"
    @ 7,35 SAY " DOLARES "
  ELSE
    @ 7,35 SAY "         "
  ENDIF
  SET COLOR TO &C_VENTANA
  MEMOEDIT(DESC,10,05,18,75,,F,,32)


  ***** AVISO DEL ESTADO ACUAL DEL REGISTRO *******
  SET COLOR TO W/B
  @ 24,00
  @ 24,01 SAY "REG. :"
      @ 24,08 SAY ALLTRIM(STR(RECN())) + " DE " + ALLTRIM(STR(RECC()))
  DO CASE
    CASE RECC() = 1
      @ 24,23 SAY " 1 SOLO REGISTRO "
    CASE RECC() = 0
```

```
    @ 24,23 SAY " ARCHIVO SIN DATOS  "
  CASE REA = REF
    *       ?? CHR(7)
    @ 24,23 SAY " FIN DE ARCHIVO     "
  CASE REA = REI
    *       ?? CHR(7)
    @ 24,23 SAY " PRINCIPIO DE ARCHIVO"
  ENDC
  IF DELETED()
   SET COLOR TO W + */B
   @ 24,44 SAY " MARCADO "
  ENDIF
  SET COLOR TO N/BG
  ************ SELECCION DE LA TECLA DE CONTROL ********

  SET COLOR TO W/B
  @ 23,00
  @ 23,00 SAY "ALTA BUSCAR CAMBIO HOJA TRAER MARCAR
SUPRIMIR "
                                                        *
0123456789012345678901234567890123456789012345678901234567890123
*          1     2     3     4     5
  SET COLOR TO W + /B
  @ 23,00 SAY "A"
  @ 23,05 SAY "B"
  @ 23,12 SAY "C"
  @ 23,19 SAY "H"
  @ 23,24 SAY "T"
  @ 23,30 SAY "M"
  @ 23,37 SAY "S"
  @ 24,55 SAY CHR(25) + CHR(24) + " PGUP PGDN HOME END
ESC"
  @ 23,79 SAY "
  ********** ACION DE LA TECLA PRESIONADA **********
  T = 0
  @ 23,78 SAY ""
  DO WHIL T = 0
   T = INKEYTRAP()
  ENDDO  *
  SET COLOR TO &C_CAJA
  @ 23,00 CLEAR
  DO CASE
     CASE T = 1     &&    TECLA    HOME       1 = DB3    26
= DB4
       GO REI
       REA = RECNO()
    CASE T = 6     &&    TECLA    END        6 = DB3    2 = DB4
       GO REF
       REA = RECNO()
    CASE T = 18    &&    TECLA    PGUP
       S = 0
       DO WHILE RECN()REI
         SKIP -1
         REA = RECNO()
         S = S + 1
         IF S = 10
           EXIT
         ENDIF
       ENDDO
    CASE T = 3     &&    TECLA    PGDN
       S = 0
       DO WHILE RECN()REF
         SKIP
         REA = RECNO()
         S = S + 1
         IF S = 10
           EXIT
         ENDIF
       ENDDO
       IF EOF()
         SKIP -1
         REA = RECNO()
       ENDI
    CASE T = 24    &&    TECLA    FLECHA ABAJO
      IF REA = REF
        LOOP
      ELSE
        SKIP 1
        REA = RECNO()
      ENDIF
    CASE T = 5     &&    TECLA    FLECHA ARRIBA
      IF REA = REI
        LOOP
      ELSE
        SKIP -1
        REA = RECNO()
      ENDIF
    CASE T = 27    &&    TECLA    ESC
      EXIT
    CASE CHR(T)$"BB"
      ********** BUSQUEDA SEEK FIND LOCATE ********
      BUSC = SPACE(14)
      MFACT = FACTURA
      MTIPO = TIPO
      MNUMERO = NUMERO
      @  2,27 SAY MFACT PICT "99999999"
      @  7, 2 GET MTIPO PICT "@!" VALID MTIPO$"FEMANM"
      @  7, 5 GET MNUMERO PICT "9999"
      READ
      IF LASTKEY() = 27
        LOOP
      ELSE
        BUSC = ALLTRIM(STR(MFACT)) + MNUMERO + MTIPO
        BUSC = TRIM(BUSC)
        FIND &BUSC
        IF FOUND()
          REA = RECNO()
        ENDI
      ENDI
    CASE CHR(T)$"AACC"
      ******** CAPTURA DE DATOS DEL REGISTO ACUTUA***
      DO WHIL .T.
        SET COLOR TO &C_ENTRA
        IF CHR(T)$"AA"
          GO REF
          MFACT = FACTURA
          MTIPO = TIPO
          @  1,67 SAY RECC() + 1 PICT "999,999,999"
          @  2,27 SAY MFACT PICT "99999999"
          @  7, 2 GET MTIPO PICT "@!" VALID MTIPO$"FEMANM"
          READ
          IF LASTKEY() = 27
            EXIT
          ENDIF
          BUSC = ALLTRIM(STR(MFACT))
          BUSC = TRIM(BUSC)
          FIND &BUSC
          DO WHILE MFACT = FACTURA
            MNUMERO = NUMERO
            SKIP
          ENDDO
```

```
            MNUMERO = RIGHT("0000" + RTRIM(LTRIM(STR(VAL
(MNUMERO) + 1))),4)
            @ 7, 5 GET MNUMERO PICT "9999"
            READ
            IF LASTKEY() = 27
              EXIT
            ENDIF
            SET EXACT ON
              BUSC = ALLTRIM(STR(MFACT)) + MNUMERO + MTI
PO
            FIND &BUSC
            SET EXACT OFF
            IF FOUND()
              OP = " "
              SET COLOR TO &C_CAJA
              @ 23,00 CLEAR
                @ 23,00 SAY " NUMERO EN PLANO REPETIDO
:" + STR(FACTURA) + "  " + TIPO + "  " + NUMERO + "
REG:" + STR(RECC())
              @ 24,10 SAY "DESEAS CONTINUAR S/N "
              @ 24,37 GET OP PICT "@!" VALID OP$"SN"
              READ
              IF OP = "N" .OR. LASTKEY() = 27
                LOOP
              ENDIF
            ENDI
            SELE 4
            SEEK MFACT
            IF .NOT. FOUND()
              OP = " "
              SET COLOR TO &C_CAJA
              @ 23,00 CLEAR
              @ 23,10 SAY "NO HAY  FACTURA  -"
              @ 23,32 SAY MFACT
              @ 24,10 SAY "DESEAS CONTINUAR  S/N "
              @ 24,37 GET OP PICT "@!" VALID OP$"SN"
              READ
              IF OP = "N"
                SELE 3
                LOOP
              ENDIF
            ENDIF
            SELE 3
            IF LASTKEY() = 27
              EXIT
            ENDIF
            APPEN BLANK
            REPLACE FACTURA WITH MFACT
            REPLACE TIPO WITH MTIPO
            REPLACE NUMERO WITH MNUMERO
            REA = RECNO()
            REF = RECNO()
          ENDI
          REA = RECNO()
          SET COLOR TO &C_ENTRA
          @  1,67 SAY RECNO() PICT "999,999,999"
          DO CASE
            CASE CHR(T)$"CC"
              @ 2,27 SAY FACTURA PICT "99999999"
              @ 7, 2 GET TIPO PICT "@!" VALID TIPO$"FEMVN"
              @ 7, 5 GET NUMERO PICT "9999"
            CASE CHR(T)$"AA"
              DO TRAER
              @  1,67 SAY RECNO() PICT "999,999,999"
              @  2,27 SAY FACTURA PICT "99999999"
              @  2,67 SAY RECCOUNT() PICT "999,999,999"
              @ 7, 2 SAY TIPO PICT "@!"
              @ 7, 5 SAY NUMERO PICT "9999"
          ENDCASE
          READ
          SET COLOR TO &C_CAJA
          @ 23,00
            @ 23,00 SAY "    CTRL W  =  GRABA Y SALE
ESC  =  SALIR SIN GRABAR "
          SET COLOR TO &C_VENTANA
                                    REPLACE DESC WITH
MEMOEDIT(DESC,10,05,18,75,,T,,70)
          SET COLOR TO &C_CAJA
          @ 23,00
          SET COLOR TO &C_ENTRA
          @  7,47 GET CANTIDAD  PICT "9,999"
          @  7,53 GET PRECIO    PICT "9999,999.99"
          READ
          MIMPORTE = CANTIDAD*PRECIO
          REPLACE IMPORTE WITH MIMPORTE
          @  7,65 SAY IMPORTE   PICT "9999,999.99"
          @  8,53 GET COSTO     PICT "9999,999.99"
          READ
          REPLACE IMPO_COST WITH CANTIDAD*COSTO
          @  8,65 SAY IMPO_COST PICT "9999,999.99"
          @ 20,20 GET ORD_COMP PICT "@!"
          @ 21,10 GET STATUS   PICT "@!"
          READ
          @ 23,00 CLEAR
          @ 23,00 SAY "ESPERA UN MOMENTO..."
          REA = RECNO()
          MFACT = FACTURA
          MBUSCA = ALLTRIM(STR(FACTURA))
          GO TOP
          FIND &MBUSCA
          IF FOUND()
            REI = RECN()
            DO WHILE FACTURA = MFACT .AND. .NOT. EOF()
              REF = RECNO()
              SKIP
            ENDDO
          ELSE
            SET COLOR TO &C_RESALTA
            @ 23,00
              @ 23,00 SAY "NO SE LOCALIZO REFERENCIA...
CAMBIO DE APUNTADORES."
              INKEYTRAP(0)
              GO TOP
              REI = RECNO()
              GO BOTT
              REF = RECN()
            ENDI
            GO REA
            IF CHR(T)$"CC"
              EXIT
            ELSE
              T = ASC("A")
            ENDI
          ENDD
        CASE CHR(T)$"MM"
      ********** SECCION DELETE Y RECALL **********
          @ 23,00 CLEAR
          @ 23,02 SAY "ESPERA UN MOMENTO..."
          IF DELETED()
            RECALL
          ELSE
            DELETE
```

```
        ENDIF
      CASE CHR(T)$"HH"
 ********** VISUALIZACION EN FORMA TABULAR *******
      * " MPH = SAVESCREEN(01,00,24,79)
        SET COLOR TO &C_CAJA
        MFACT = FACTURA
        @ 24,00
" + CHR(29) + CHR(28) + CHR(27) + CHR(24) + " HOME END PGDN
PGUP          ESC O ENTER = SALIR "
        SET COLOR TO &C_TEXT
        DBEDIT(1,0,23,79)
        RESTSCREEN(01,00,24,79,MPH)
        RELEASE MPH
        IF FACTURA = MFACT
          REA = RECN()
        ENDI
        LOOP
      CASE CHR(T)$"TT"
        DO TRAER
      CASE CHR(T)$"SS"
        IF &NIVEL1
          LOOP
        ENDI
 ********** SECCION DELETE Y RECALL **********
        @ 23,00 CLEAR
        @ 23,00 SAY " ESPERA UN MOMENTO... "
        @ 24,00 SAY "                    ELIMINANDO REGISTROS
MARCADOS Y RECONTRUYENDO INDEX."
        PACK
        GO TOP
        FIND &MREFA
        IF FOUND()
          @ 23,00 CLEAR
          @ 23,00 SAY "ESPERA UN MOMENTO GENERANDO
APUNTADORES..."
          RET = RECN()
          REA = RECN()
          DO WHIL FACTURA = VAL(&MREFA)
            REF = RECN()
            SKIP
        * ENDO
        ELSE
          ?? CHR(7)
          @ 23,00 CLEAR
          @ 23,00 SAY "  AVISO ...: LA FACTURA HA PERDIDO
TODOS LOS PRODUCTOS DEL PEDIDO... "
          @ 24,00 SAY "  OPCION...: MARCAR Y SUPRIMIR EL
FACTURA    PRESIONA ALGUNA TECLA... "
          INKEYTRAP(0)
          EXIT
        ENDI
      OTHER
        *       ?? CHR(7)
        T = 0
        LOOP
    ENDC
  ENDO
  RESTSCREEN(00,00,24,79,MP2)
  RELEASE MP2
  RETURN
```

MOVE.PRG

```
*****************************************************
* PROGRAMA.............:
* NOTAS................: MOVIMIENTO A
* BASE DE DATOS.......:
* INDEX ...............:
* SELECT ..............:
* FECHA ...............:
* VERSION .............: 5.0        (ADAPTACION PARA CLIPPER5)
* REALIZADO POR       : ARTURO CROTTE FRANC      *
* INCLUYE MODULOS:
*     : PANTALLA,MUESTRA,CAPTA,VALIDA,BROWSE
*
*****************************************************
MPMI = SAVESCREEN(00,00,24,79)

SELE 1
USE ... INDEX ... VIA "DBFNDX"
STORE 0 TO REA,REI,REF,T
***** FORMACION DE PANTALLA DE VISUALIZACION *******
DO PRE_PAN WITH 0
******** @ SAY 'LETREROS DE FORMATO BASE' **********
* PANTALLA

*****************************************************
GO BOTT
REF = RECN()
GO TOP
REI = RECN()
REA = RECN()
*************** LOOP - PRINCIPAL ********************
DO WHIL .T.
  IF RECC() # 0
    GO REA
  ENDI

  ********** VISUSUZIACION DE L REGISTRO ACTUAL *****
  SET COLOR TO &C_ENTRA
* MUESTRA
  ******** AVISO DEL ESTADO ACUAL DEL REGISTRO *******
  DO ESTADO
  ************** SELECCION DE LA TECLA DE CONTROL ***
  SET COLOR TO &C_CAJA
  @ 23, 00
    @ 23, 00 SAY ' ALTA BUSCAR CAMBIO TABULAR MARCA SUPRIMIR'

01234567890123456789012345678901234567890123456789012345678901234567890
    *          1    2    3    4    5    6
  SET COLOR TO &C_RESALTA
  @ 23,01 SAY 'A'
  @ 23,08 SAY 'B'
  @ 23,13 SAY 'C'
  @ 23,20 SAY 'T'
  @ 23,26 SAY 'M'
  @ 23,34 SAY 'S'
  @ 24,60 SAY CHR(25) + CHR(24) + ' HOME END O ESC'
  SET COLOR TO &C_TEXT

  ********** ACCION DE LA TECLA PRESIONADA ************
  T = 0
  DO WHIL T = 0
    T = INKEY()
  ENDD
  @ 23,00
  DO CASE
    CASE T = 1    .AND.   TECLA  HOME   1 = DB3   26 = DB4
      GO TOP
      REA = RECN()
      REI = RECN()
    CASE T = 6    .AND.   TECLA  END    6 = DB3   2 = DB4
      GO BOTT
      REA = RECN()
      REF = RECN()

    CASE T = 18   .AND.   TECLA  PGUP
      S = 0
      DO WHIL .NOT. BOF()
        SKIP -1
        REA = RECN()
        S = S + 1
        IF S = 10
          EXIT
        ENDI
      ENDD
    CASE T = 3    .AND.   TECLA  PGDN
      S = 0
      DO WHIL .NOT. EOF()
        SKIP
        REA = RECN()
        S = S + 1
        IF S = 10
          EXIT
        ENDI
      ENDD
      IF EOF()
        SKIP -1
        REA = RECN()
      ENDI
    CASE T = 24   .AND.   TECLA  FLECHA ABAJO
      IF REA = REF
        LOOP
      ELSE
        SKIP 1
        REA = RECN()
      ENDI
    CASE T = 5    .AND.   TECLA  FLECHA ARRIBA
      IF REA = REI
        LOOP
      ELSE
        SKIP -1
        REA = RECN()
      ENDI

    CASE T = 27   .AND.   TECLA  ESC
      EXIT
    CASE CHR(T) $ 'BB'
      ***** BUSQUEDA SEEK FIND LOCATE *********
      SET COLOR TO &C_ENTRA
* VALIDA
      READ
      IF LASTKEY() = 27
        LOOP
      ENDI
```

```
        *       LOCATE FOR MBUSCA&CAMPO
        *       SEEK MBUSCA
        *       FIND &MBUSCA
       IF FOUND()
         REA = RECN()
       ENDI
     CASE CHR(1)$'AADC'
     ********* CAPTURA DE DATOS DEL REGISTO ACUTUAL *
       SET COLOR TO &C_ENTRA
       IF 'CHR(1)$'AA'
  * VALIDA
         READ
        IF LASTKEY() = 27
          LOOP
        ENDI
        *       LOCATE FOR MBUSCA&CAMPO
        *       SEEK MBUSCA
        *       FIND &MBUSCA
        IF FOUND()
          ?? CHR(7)
          SET COLOR TO &C_RESALTA
          @ 23,00 CLEAR
            @ 23,10 SAY ' !!! ATENCION !!! IDENTIFICADOR
REPETIDO.'
          @ 24,10 SAY ' PRESIONA CUAQUIER TECLA PARA
CONTINUAR...'
          INKEY(0)
          REA = RECN()
          LOOP
        ENDI
        APPE BLANK
        *       REPL      WITH MBUSCA
        SET COLOR TO &C_CAJA
        @ 23,00
          @ 23,10 SAY ' AGREGANDO UN REGISTRO AL
ARCHIVO.          (ALTA) '
        SET COLOR TO &C_ENTRA
       ENDI
       REA = RECN()
       ********** GET CAMPOS DE CAPTURA **********
  * CAPTA
        READ
       GO TOP
       REI = RECN()
       GO BOTT
       REF = RECN()
       GO REA
      LOOP
     CASE CHR(1)$'SS'
       ******** SECCION PACK Y REINDEX **********
       SET COLOR TO &C_CAJA
       @ 23,00
         @ 23,10 SAY ' ESPERA UN MOMENTO ELIMINANDO
REGISTROS MARCADOS '
       PACK
       GO BOTT
       REF = RECN()
       GO TOP
       REI = RECN()
       REA = RECN()
       LOOP
     CASE CHR(1)$'MM'
       ****** SECCION DELETE Y RECALL ************
       IF DELE()
         RECA
        ELSE
         DELE
       ENDI
       LOOP
     CASE CHR(1)$'TT'
       ******** VISUALIZACION EN FORMA TABULAR ******
       MPM2 = SAVESCREEN(01,00,24,79)
       SET COLOR TO &C_CAJA
       @ 24,00
                                  @ 24,00 SAY ' USE
' + CHR(26) + CHR(25) + CHR(27) + CHR(24) + ' HOME END PGDN
PGUP                    ESC = SALIR '
       SET COLOR TO &C_TEXT
       BROWSE(1,0,23,79)
       RESTSCREEN(01,00,24,79,MPM2)
       RELE MPM2
       REA = RECN()
       LOOP
     OTHER
       T = 0
       LOOP
   ENDC
 ENDD
 @ 23,1 SAY ' '
 RESTSCREEN(00,00,24,79,MPM1)
 RELE MPM1
 USE
 RETU
 * ECU(MOVE)
```

MENUPERS.PRG

```
*-------------------------------------------------*
* PROGRAMA      : MENUPERS.PRG                    *
* REALIZADO POR ARTURO CROTTE FRANCO.             *
* BASES DE DATOS :                                *
*                                                 *
* FECHA         : 09-MAR-92                       *
* NOTAS         :                                 *
*                                                 *
*          APLICACION PARA CLIPPER                *
*                                                 *
*-------------------------------------------------*
MPAN = SAVESCREEN(0,0,24,79)

* STARTCOLOR = SETCOLOR()
* SET KEY 301 TO DOQUIT
* SET KEY 274 TO DOSSHELL
* SET PROC TO SFRO
* DO ENTRA
IF MNIVEL = 0
  CLOSE ALL
  RESTSCREEN(00,00,24,79,MPAN)
  @ 23,00 SAY ""
  QUIT
ENDI
  SET WRAP ON                    && MOVIMIENTO ROTATORIO
EN MEUS
  SET MESS TO 24 CENTER          && DESPLIEGA MENSAJES
EN LINEA 24
  CLEAR
  PADS = 1                       && VARIABLE DE MEMORIA MENU
PRINCIPAL
  POPU = 1                       && VARIABLE DE MEMORIA MENU
POPUP
  KEYBOARD CHR(13)               && ACTIVAR MENU DESDE
ENTRADA
  DO WHIL .T.
    IF LASTKEY() = 27
      SET COLOR TO &C_TEXT
      @ 1,0 CLEAR TO 23,0
    ENDI
    SET COLOR TO &C_CAJA
    @ 00,00
    @ 24,00
    @ 00,03 PROMPT "AAAAAAAAA"
    @ 00,15 PROMPT "BBBBB"
    @ 00,24 PROMPT "CCCCCCC"
    @ 00,34 PROMPT "DDDDDDDD"
    @ 00,44 PROMPT "EEEEEEEE"
    @ 00,54 PROMPT "FFFFFFFFFF"
    @ 00,72 PROMPT "TERMINAR"
    MENU TO PADS
    IF PADS # 0
      DO SELE_PADS WITH PADS
    ENDI
    LOOP
  ENDD

  PROC SELE_PADS
  PARA PAD
    * ASIGNA FLECHAS PARA SIGIENTE MENU ISQUIERDA O
DERECHA
    SET KEY 19 TO FLECHA_IZQ
    SET KEY  4 TO FLECHA_DER
    RELEASE POPU                 && LIMPIA LA VARIABLE POPU
    SET COLOR TO &C_TEXT
     @ 01,00 CLEAR TO 23,79
    SET COLOR TO &C_CAJA
    DO CASE
      CASE PAD = 1
        @ 01,03 CLEAR TO 05,19
        @ 01,03 TO 05,19
        @ 02,05 PROM 'MOVIMIENTOS'
        @ 03,05 PROM 'ORDENAR'
        @ 04,05 PROM 'SUPRIMIR'
      CASE PAD = 2
        @ 01,15 CLEAR TO 05,30
        @ 01,15 TO 05,30
        @ 02,16 PROM 'MOVIMIENTOS'
        @ 03,16 PROM 'ORDENAR'
        @ 04,16 PROM 'SUPRIMIR'
      CASE PAD = 3
        @ 01,24 CLEAR TO 05,38
        @ 01,24 TO 05,38
        @ 02,26 PROM 'MOVIMIENTOS'
        @ 03,26 PROM 'ORDENAR'
        @ 04,26 PROM 'SUPRIMIR'
      CASE PAD = 4
        @ 01,34 CLEAR TO 05,49
        @ 01,34 TO 05,49
        @ 02,36 PROM 'MOVIMIENTOS'
        @ 03,36 PROM 'ORDENAR'
        @ 04,36 PROM 'SUPRIMIR'
      CASE PAD = 5
        @ 01,44 CLEAR TO 05,58
        @ 01,44 TO 05,58
        @ 02,46 PROM 'MOVIMIENTOS'
        @ 03,46 PROM 'ORDENAR'
        @ 04,46 PROM 'SUPRIMIR'
      CASE PAD = 6
        @ 01,54 CLEAR TO 05,68
        @ 01,54 TO 05,68
        @ 02,56 PROM 'MOVIMIENTOS'
        @ 03,56 PROM 'ORDENAR'
        @ 04,56 PROM 'SUPRIMIR'
      CASE PAD = 7
        @ 01,68 CLEAR TO 05,79
        @ 01,68 TO 05,79
        @ 02,70 PROM 'CONTINUA'
        @ 03,68 SAY 'MMMMMMMMMM'
        @ 04,70 PROM 'TERMINAR'
    ENDC

    MENU TO POPU
    * ASIGNA FLECAHAS PARA USO NORMAL
    SET KEY 19 TO
    SET KEY  4 TO
    DO CASE
      CASE PAD = 1
        *
      CASE PAD = 2
        *
      CASE PAD = 3
        *
      CASE PAD = 4
        *
      CASE PAD = 5
        *
```

```
  CASE PAD = 6
    *
  CASE PAD = 7
     DO CASE
       CASE POPU = 1
          KEYBOARD CHR(13)
       CASE POPU = 2
          RESTSCREEN(0,0,24,79,MPAN)
          @ 23,00 SAY ''
          CLOSE ALL
          QUIT
     ENDC
ENDC
RETURN

PROC FLECHA_IZQ
PARA PN, PL, RV
* RETURN, UP ARROW, RETURN
KEYBOARD CHR(27) + CHR(5) + CHR(13)
RETURN


PROC FLECHA_DER
PARA PN, PL, RV
* RETURN, RIGHT ARROW, RETURN
KEYBOARD CHR(27) + CHR(4) + CHR(13)
RETURN
```

PPRO.PRG

```
*************************************************************
*PROCEDIMIENTOS Y FUNCIONES GENERALES                       *
*************************************************************
**** PROCEDIMIENTO DE FECHA EN STRING ********** **
PROC SFECHA
DO CASE
  CASE MONTH(MFECHA) = 1
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE ENERO DE
' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 2
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE FEBRERO DE
' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 3
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE MARZO DE
' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 4
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE ABRIL DE
' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 5
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE MAYO DE
' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 6
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE JUNIO DE
' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 7
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE JULIO DE
' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 8
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE AGOSTO DE
' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 9
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE SEPTIEMBRE
DE ' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 10
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE OCTUBRE DE
' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 11
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE NOVIEMBRE
DE ' + RIGHT(STR(YEAR(MFECHA)),4)
  CASE MONTH(MFECHA) = 12
       MES = RIGHT(STR(DAY(MFECHA)),2) + ' DE DICIEMBRE DE
' + RIGHT(STR(YEAR(MFECHA)),4)
ENDCASE
  MSFECHA = MES
RETU


*************************************************************
*************************************************************
*************************************************************
*
*  CONSULTA DE CLIENTES
*
PROCEDURE CON_CLIE
MSCC = SAVESCREEN(00,00,24,79)
SET COLOR TO &C_CAJA
SET KEY -8 TO
MNUM = 0
@ 23,00
@ 23,00 SAY ' FLECHAS = SELECCIONAR  ENTER = ACEPTAR
ESC = ABANDONAR '
@ 6,05 CLEAR TO 21,75
@ 6,05 TO 21,75 DOUBLE
@ 6,30 SAY 'C L I E N T E S'
MNUM = ACHOICE(7,13,20,73,MCLIENTES)
@ 23,00
RESTSCREEN(00,00,24,79,MSCC)
RELEASE MSCC
IF MNUM = 1 .AND. MNUM
  MNOMBRE = MCLIENTES[MNUM]
ENDIF
SET KEY -8 TO CON_CLIE
SET COLOR TO &C_ENTRA
RETU
*************************************************************
*************************************************************
*************************************************************
*
*  CONSULTA DE PROVEEDORES
*
PROCEDURE CON_PROV
MSCC = SAVESCREEN(00,00,24,79)
SET COLOR TO &C_CAJA
SET KEY -8 TO
MNUM = 0
@ 23,00
@ 23,00 SAY ' FLECHAS = SELECCIONAR  ENTER = ACEPTAR
ESC = ABANDONAR '
@ 6,05 CLEAR TO 21,75
@ 8,05 TO 21,75 DOUBLE
@ 8,28 SAY 'P R O V E E D O R E S'
MNUM = ACHOICE(7,13,20,73,MPROVS)
@ 23,00
RESTSCREEN(00,00,24,79,MSCC)
RELEASE MSCC
IF MNUM = 1 .AND. MNUM
  MNOMBRE = MPROVS[MNUM]
ENDIF
SET KEY -8 TO CON_PROV
SET COLOR TO &C_ENTRA
RETU
*************************************************************
*************************************************************
*************************************************************
*
*  CONSULTA DE PRODUCTOS BASICOS
*
PROCEDURE CON_BAS
MSCB = SAVESCREEN(00,00,24,79)
SET COLOR TO &C_RESALTA
@ 23,00
  @ 23,00 SAY ' ESPERA UN MOMENTO ACTUALIZANDO
CONSULTA DE PRODUCTOS BASICOS.'
  ************************
  *** TABLA DE PRODUCTOS BASICOS  ***
  ************************
  MAREA = STR(SELE(),1)
  SELE 2
  USE PBASICOS INDEX PBASICOS VIA 'DBFNDX'
  TOTAL = RECC()
  RELEASE MBASICOS
  DECLARE MBASICOS[TOTAL]
  MPB = 1
  GO TOP
  DO WHILE .NOT.EOF()
                MBASICOS[MPB] = TIPO + '-' + LISTA + '
' + MEMOLINE(DESC,45,1) + ' ' + TRAN(PRECIO,"9999,999,999")
    SKIP
    MPB = MPB + 1
  ENDDO
  SELE &MAREA
```

```
RELEASE MAREA
SET COLOR TO &C_CAJA
SET KEY -6 TO
MNUM = 0
@ 23,00
@ 23,00 SAY ' FLECHAS = SELECCIONAR  ENTER = ACEPTAR  ESC = ABANDONAR '
@ 8,05 CLEAR TO 21,75
@ 8,05 TO 21,75 DOUBLE
@ 8,29 SAY ' PRODUCTOS BASICOS '
MNUM = ACHOICE(7,07,20,73,MBASICOS)
@ 23,00
RESTSCREEN(00,00,24,79,MSCB)
RELEASE MSCB
IF MNUM = 1 .AND. MJUM
  MTIPO = LEFT(MBASICOS[MNUM],2)
  MLISTA = SUBSTR(MBASICOS[MNUM],4,4)
ENDIF
SET KEY -8 TO CON_BAS
SET COLOR TO &C_ENTRA
RETU

*****************************************************
*****************************************************
*****************************************************

PROCEDURE CRECE
PARAMETER Y1,X1,Y2,X2,COL,MARC
L = X2-X1 .
A = Y2-Y1
ORIX = X1 + INT((L/2))
ORIY = Y1 + INT((A/2))
FINX = ORIX
FINY = ORIY
DO WHILE .T.
  IF ORIX>1
    ORIX = ORIX-1
  ENDIF
  IF ORIYY1
    ORIY = ORIY-1
  ENDIF
  IF FINX2
    FINX = FINX + 1
  ENDIF
  IF FINYY2
    FINY = FINY + 1
  ENDIF
  SET COLOR TO &COL
  @ ORIY,ORIX CLEAR TO FINY,FINX
  IF MARC
    @ ORIY,ORIX TO FINY,FINX
  ENDIF
  IF ORIX = X1 .AND. FINX = X2 .AND. ORIY = Y1 .AND. FINY = Y2
    EXIT
  ENDIF
ENDDO
*
*---PONE SOMBRA
*
SOMBRA = (FINX + 1)-(ORIX)
SOMBRA2 = ORIY + 1
SET COLOR TO N + /N
DO WHILE SOMBRA2 Y + 1
  @ SOMBRA2,ORIX + 1 SAY REPLICATE (CHR(176),SOMBRA)
  SOMBRA2 = SOMBRA2 + 1
ENDDO
*
*---PONE CUADRO
*
SET COLOR TO &COL
@ ORIY,ORIX CLEAR TO FINY,FINX
IF MARC
  @ ORIY,ORIX TO FINY,FINX
ENDIF
RETURN

*****************************************************
*****************************************************
*****************************************************

PROC TRAER
RDA = REA
RDI = REI
RDF = REF
SELE 2
SET COLOR TO &C_RESALTA
@ 23,00
  @ 23,00 SAY' TECLEA TIPO Y N#MERO EN CATLOGO DE PRODUCTOS BASICOS     F8 CONSULTA'
BUSC = SPACE(6)
PUBLIC MTIPO,MLISTA
MTIPO = ' '
MLISTA = '  '
MPASOMEMO = ''
SET COLOR TO &C_ENTRA
SET KEY -8 TO CON_BAS
@ 7,15 GET MTIPO PICTURE '@!'
@ 7,18 GET MLISTA PICTURE '9999'
READ
SET KEY -8 TO
IF LASTKEY() = 27
  SELE 3
  T = 0
  SET COLOR TO &C_TEXT
  @ 7,15 SAY '      '
  RETU
ENDIF
BUSC = MTIPO + MLISTA
FIND &BUSC
IF .NOT. FOUND()
  SET COLOR TO &C_CAJA
  @ 23,00
    @ 23,00 SAY ' NO LOCALIZADO... CONSULTAMOS EN CATLOGO DE PRODUCTOS BASICOS'
  INKEYTRAP(5)
  PUBL &REL,MPASOMEMO,MPRECIO
  MPASOMEMO = ''
  MPRECIO = 0
  MREL = 1
  DO PBASMOVE
ENDIF
MPASOMEMO = DESC
MPRECIO = PRECIO
SELE 3
REA = RDA
REF = RDF
REI = RDI
REA = RECN()
REPLACE DESC WITH MPASOMEMO
REPLACE PRECIO WITH MPRECIO
REPLACE IMPORTE WITH CANTIDAD*PRECIO
RELE MPASOMEMO
```

```
RELE MPRECIO
RELE MREL
T = 0
SET COLOR TO &C_TEXT
@ 7,15 SAY '      '
RETU

*****************************************************
*****************************************************
*****************************************************
PROC TRAERBAS
BUSC = SPACE(8)
MTIPO = ' '
MUSTA = '   '
@ 8,02 GET MTIPO PICTURE "@!" VALID MTIPOS"MINFE3M"
@ 8,05 GET MUSTA PICTURE "9999"
READ
IF LASTKEY() = 27
  RETU
ENDIF
BUSC = MTIPO + MUSTA
FIND &BUSC
IF FOUND()
  MPASOMEMO = ''
  MPASOMEMO = DESC
  GO REA
  REPLACE DESC WITH MPASOMEMO
  RELE MPASOMEMO
  T = 0
 ELSE
  SET COLOR TO &C_CAJA
  @ 23,00 CLEAR
    @ 23,00 SAY "BUSQUEDA  SIN  EXITO ... SI TU CLAVE FUE
CORRECTA SUGIERO RECONTRUIR INDICES"
    @ 24,00 SAY "PRESIONA CUALQUIER TECLA PARA
CONTINUAR "
  INKEYTRAP(0)
ENDIF
RETU

*****************************************************
*****************************************************
*****************************************************
PROCEDURE SALIDA
SET COLOR TO
CLEA
RESTSCREEN(0,0,24,80,MPAN)
DECRECE(0,0,24,79,C_TEXT,.T.)
CUADRITO(12,40,24,0)
@ 24,00
SET COLOR TO W+/N
SET COLOR TO W/N
?
SET COLOR TO
RETURN



*****************************************************
*****************************************************
*****************************************************



FUNCTION CUADRITO
PARAMETER INIY,INIX,Y1,X1
PANTA = SAVESCREEN(0,0,24,79)
POSY = INIY
POSX = INIX
XYA = .F.
YYA = .F.
HORIZ = REPLICATE(CHR(219),5)
SET COLOR TO
DO WHILE .T.
  IF INYY1
    @ POSY,POSX SAY HORIZ
    @ POSY-1,POSX SAY HORIZ
  ELSE
    @ POSY,POSX SAY HORIZ
    @ POSY + 1,POSX SAY HORIZ
  ENDIF
  IF POSYY1
    IF INYY1
      POSY = POSY-1
    ELSE
      POSY = POSY + 1
    ENDIF
  ELSE
    YYA = .T.
  ENDIF
  IF POSXX1
    IF INDX1
      POSX = POSX-1
    ELSE
      POSX = POSX + 1
    ENDIF
  ELSE
    XYA = .T.
  ENDIF
  RESTSCREEN(0,0,24,79,M-PANTA)
  IF YYA .AND. XYA
    EXIT
  ENDIF
ENDDO
RETURN .T.

*****************************************************
*****************************************************
*****************************************************
FUNCTION DECRECE
PARAMETER Y1,X1,Y2,X2,COL,MARC
PANPRIN = SAVESCREEN(0,0,24,79)
L= X2-X1
A = Y2-Y1
TERMX = X1 + INT((L/2))
TERMY = Y1 + INT((A/2))
ORIX = X1
ORIY = Y1
FINX = X2
FINY = Y2
DO WHILE .T.
  IF ORIX

    ORIX = ORIX + 1
  ENDIF
  IF ORIY
Y
    ORIY = ORIY + 1
  ENDIF
  IF FINXTERMX
    FINX = FINX-1
  ENDIF
  IF FINYTERMY
    FINY = FINY-1
```

```
   ENDIF
   RESTSCREEN (0,0,24,79,M-PANPR3)
   SETCOLOR(COL)
   @ ORY,ORX CLEAR TO FINY,FINX
   IF MARC
     @ ORY,ORX TO FINY,FINX
   ENDIF
   IF ORX = TERMX .AND. FINX = TERMX .AND. ORY = TERMY
.AND. FINY = TERMY
     EXIT
   ENDIF
 ENDDO
 RESTSCREEN (0,0,24,79,M-PANPR4)
 RETU .T.

*************************************************
*************************************************
*************************************************
PROC REGPEDI
  * PROCEDIMIENTO PARA REGRESAR PEDIPROD SIN
PROCESO DE RECONTRUCCION
 USE
 COPY FILE RESPALDO.DBF TO PEDIPROD.DBF
 COPY FILE RESPALDO.DBT TO PEDIPROD.DBT
 COPY FILE RESPALDO.NDX TO PEDIPROD.NDX
 USE PEDIPROD INDEX PEDIPROD VIA 'DBFNDX'
 RETU

*************************************************
*************************************************
*************************************************

PROC REGFACT
  * PROCEDIMIENTO PARA REGRESAR FACTURAS SIN
PROCESO DE RECONTRUCCION
 USE
 COPY FILE RESPALDO.DBF TO PFACPROD.DBF
 COPY FILE RESPALDO.DBT TO PFACPROD.DBT
 COPY FILE RESPALDO.NDX TO PFACPROD.NDX
 USE PFACPROD INDEX PFACPROD VIA 'DBFNDX'
 RETU

*************************************************
*************************************************
*************************************************
PROC REGPRES
  * PROCEDIMIENTO PARA REGRESAR PEDIPROD SIN
PROCESO DE RECONTRUCCION
 USE
 COPY FILE RESPALDO.DBF TO PDETALLE.DBF
 COPY FILE RESPALDO.DBT TO PDETALLE.DBT
 COPY FILE RESPALDO.NDX TO PDETALLE.NDX
 USE PDETALLE INDEX PDETALLE VIA 'DBFNDX'
 RETU
*************************************************
*************************************************
*************************************************

PROC REGREMI
  * PROCEDIMIENTO PARA REGRESAR REMISION SIN
PROCESO DE RECONTRUCCION
 USE
 COPY FILE RESPALDO.DBF TO PREMPROD.DBF
 COPY FILE RESPALDO.DBT TO PREMPROD.DBT
 COPY FILE RESPALDO.NDX TO PREMPROD.NDX
 USE PREMPROD INDEX PREMPROD VIA 'DBFNDX'
 RETU

*************************************************
*************************************************
*************************************************

PROC REGORDC
 * PROCEDIMIENTO PARA REGRESAR ORDENES DE COMPRA
SIN PROCESO DE RECONTRUCCION
 USE
 COPY FILE RESPALDO.DBF TO PORDPROD.DBF
 COPY FILE RESPALDO.DBT TO PORDPROD.DBT
 COPY FILE RESPALDO.NDX TO PORDPROD.NDX
 USE PORDPROD INDEX PORDPROD VIA 'DBFNDX'
 RETU


*************************************************
*************************************************
*************************************************

PROC DOQUIT (PROCNAME, PROCLINE, PROCVAR)
LOCAL OLDROW,OLDCOL,SCOL_SROW, OLDCURSOR
OLDROW = ROW()
OLDCOL = COL()
SCOL = (MAXCOL()-33)/2
SROW = (MAXROW()-6)/2
 OLDSCREEN = SAVESCREEN(SROW,SCOL,SROW + 7,SCOL +
34)
OLDCOLOR = SETCOLOR('GR + /B,W + /N')
OLDCURSOR = SETCURSOR(0)

@ SROW,SCOL CLEAR TO SROW + 6,SCOL + 33
@ SROW,SCOL TO SROW + 6,SCOL + 33 DOUB
@ SROW + 1,SCOL + 2 SAY ' CONFIRMAR SALIDA '
@ SROW + 3,SCOL + 15 PROM ' NO '
@ SROW + 4,SCOL + 15 SAY '---'
@ SROW + 5,SCOL + 15 PROM ' SI '
SOMBRA(SROW,SCOL,SROW + 6,SCOL + 33)
SET KEY 301 TO
TONE(1500,1)
TONE(2000,1)
MENU TO MICHOI
SETCURSOR(OLDCURSOR)
IF MICHOI = 2
  CLOSE ALL
  SETCOLOR(STARTCOLOR)
  CLEAR
  QUIT
ENDIF
SET KEY 301 TO DOQUIT
SETCOLOR(OLDCOLOR)
 RESTSCREEN(SROW,SCOL,SROW + 7,SCOL + 34,OLDSCREE
N)
SETPOS(OLDROW,OLDCOL)
RETU

*************************************************
*************************************************
*************************************************

PROC DOSSHELL (PROCNAME, PROCLINE, PROCVAR)
LOCAL OLDROW,OLDCOL,OLDCURSOR
OLDSCREEN = SAVESCREEN(0,0,MAXROW(),MAXCOL())
OLDCOLOR = SETCOLOR(STARTCOLOR)
OLDCURSOR = SETCURSOR(1)
OLDROW = ROW()
```

```
OLDCOL = COL()
CLEAR
@ 0,0 SAY " PARA REGRESAR AL PROGRAMA TECLEE --  EXIT
?
COMSPEC = GETENV("COMSPEC")
RUN(COMSPEC)
SETCOLOR(OLDCOLOR)
RESTSCREEN(0,0,MAXROW(),MAXCOL(),OLDSCREEN)
SETCURSOR(OLDCURSOR)
SETPOS(OLDROW,OLDCOL)
* POSITION
RETU

*************************************************
*************************************************
*************************************************
FUNC MSTATUS (PRONAME, PROCLINE, PROCVAR)
MPSTAT = SAVESCREEN(00,00,24,79)
@ 0,05 CLEAR TO 22,75
@ 0,05 TO 22,75
MAREA = STR(SELE(),1)
S = 1
DO WHIL S # 20
  MS = RIGHT(STR(S),2)
  SELE &MS
  @ S,10 SAY "SELE " + MS + " : " + DBF()
  S = S + 1
ENDD
SELE &MAREA
MDBF = ""
MDBF = DBF()
  @  8,10 SAY "AREA (SELECT) ACTUAL : " + MDBF + "
( " + MAREA + ")"
  @  1,50 SAY "PROGRAMA : " + PRONAME
  @  2,50 SAY "LINEA   : " + ALLTRIM(STR(PROCLINE))
  @  3,50 SAY "VARIABLE : " + PROCVAR
  @  4,50 SAY "DISKSPACE: " + ALLTRIM(STR(DISKSPACE()))
  IF .NOT. EMPTY(MDBF)
    REA = RECN()
    @  6,50 SAY "REGISTROS: " + ALLTRIM(STR(RECC()))
    @  7,50 SAY "ACTUAL  : " + ALLTRIM(STR(RECN()))
    COUNT FOR DELE() TO MDELES
    @  8,50 SAY "MARCADOS : " + ALLTRIM(STR(MDELES))
    @  9,50 SAY "CAMPOS  : " + ALLTRIM(STR(FCOUNT()))
    @ 10,50 SAY "LONGITUD : " + ALLTRIM(STR(RECSIZE()))
    @ 11,50 SAY "DBF APROX: " + ALLTRIM(STR(DBFDSKSIZE()))
    IF ISDBT()
      @ 12,50 SAY "EXISTE ARCHIVO MEMO .DBT"
    ENDI
    @ 13,50 SAY "VEL.CPU  : " + ALLTRIM(STR(SPEED())) + "+-
4.77"
    GO REA
  ENDI
  @ 21,00 CLEAR
  @  24,20 SAY "PRESIONA CUALQUIER TECLA PARA
CONTINUAR..."
  INKEY(0)
  RESTSCREEN(00,00,24,79,MPSTAT)
  SELE &MAREA
  RELE MPSTAT,MAREA
  RETU

*************************************************
*************************************************
*************************************************
```

```
FUNC MSTRUC (PRONAME, PROCLINE, PROCVAR)
LOCAL OLDROW,OLDCOL,OLDCURSOR
IF EMPTY(DBF())
  ?? CHR(7)
  RETU
ENDIF
OLDSCREEN = SAVESCREEN(0,0,MAXROW(),MAXCOL())
OLDCOLOR = SETCOLOR(STARTCOLOR)
OLDCURSOR = SETCURSOR(1)
OLDROW = ROW()
OLDCOL = COL()

SET COLOR TO &C_VENTANA
@ 0,03 CLEAR TO 22,75
@ 0,03 TO 22,73
MAREA = STR(SELE(),1)
SELE &MAREA
MDBF = ""
MDBF = DBF()
SET COLOR TO &C_RESALTA
  @  8,10 SAY " AREA (SELECT) ACTUAL :  ( " + MAREA + ")" +"
" + DBF() + ":DBF "
SET COLOR TO &C_VENTANA
@  1,50 SAY "PROGRAMA : " + PRONAME
@  2,50 SAY "LINEA   : " + ALLTRIM(STR(PROCLINE))
@  3,50 SAY "VARIABLE : " + PROCVAR
@  4,50 SAY "DISKSPACE: " + ALLTRIM(STR(DISKSPACE()))
IF .NOT. EMPTY(MDBF)
  REA = RECN()
  @  6,50 SAY "REGISTROS: " + ALLTRIM(STR(RECC()))
  @  7,50 SAY "ACTUAL  : " + ALLTRIM(STR(RECN()))
  COUNT FOR DELE() TO MDELES
  @  8,50 SAY "MARCADOS : " + ALLTRIM(STR(MDELES))
  @  9,50 SAY "CAMPOS  : " + ALLTRIM(STR(FCOUNT()))
  @ 10,50 SAY "LONGITUD : " + ALLTRIM(STR(RECSIZE()))
  @ 11,50 SAY "DBF APROX: " + ALLTRIM(STR(DBFDSKSIZE()))
  IF ISDBT()
    @ 12,50 SAY "EXISTE ARCHIVO MEMO .DBT"
  ENDI
  IF .NOT. EMPTY(INDEXKEY())
    @ 13,50 SAY "INDEX POR: " + ALLTRIM(INDEXKEY())
  ENDI
  GO REA
ENDI
@ 23,00 CLEAR
@ 24,20 SAY "PRESIONA ENTER O ESC PARA TERMINAR..."
COPY STR..) EXTE TO TEMPSTRU
SELE 20
USE TEMPSTRU
SET KEY -5 TO
SET KEY -4 TO
DBEDIT(2,4,21,52)
SET KEY -5 TO MSTATUS()
SET KEY -4 TO MSTRUC()
ERASE TEMPSTRU.DBF
SELE &MAREA
SELE &MAREA

SETCOLOR(OLDCOLOR)
RESTSCREEN(0,0,MAXROW(),MAXCOL(),OLDSCREEN)
SETCURSOR(OLDCURSOR)
SETPOS(OLDROW,OLDCOL)
RELE OLDSCREEN,MAREA
RETU
```

```
*************************************************
*************************************************
*************************************************
FUNC SOMBRA(LS,CS,LI,CI)
RELE COL,UN
COL = SAVESCREEN(LS + 1,CI + 1,LI + 1,CI + 1)
UN = SAVESCREEN(LI + 1,CS + 1,LI + 1,CI + 1)
FOR I = 2 TO LEN(COL) STEP 2
   COL = STUFF(COL,I,CHR(8))
NEXT
FOR I = 2 TO LEN(LIN) STEP 2
   UN = STUFF(UN,I,1,CHR(8))
NEXT
RESTSCREEN(LS + 1,CI + 1,LI + 1,CI + 1,COL)
RESTSCREEN(LI + 1,CS + 1,LI + 1,CI + 1,UN)
RETU(.T.)

*************************************************
*************************************************
*************************************************
PROC CREAFONDO
* FONDO DE PANTALLA PARA MENU PRINCIPAL
SET COLOR TO &C_TEXT
@ 01,00 CLEAR TO 23,79


MFONDO = SAVESCREEN(01,00,23,79)
RETU

*************************************************
*************************************************
*************************************************
PROC FONDO
RESTSCREEN(01,00,23,79,MFONDO)
RETU

*************************************************
*************************************************
*************************************************

PROC SUENA
TONE(1500,1)
TONE(2000,1)
TONE(1500,1)
RETU
```

APÉNDICE B

PROGRAMAS DE LA APLICACIÓN DE PRESUPUESTOS

CONTENIDO DEL APÉNDICE B

P.PRG

```
*------------------------------------------------*
* PROGRAMA     : P.PRG                           *
* REALIZADO POR ARTURO CROTTE FRANCO.            *
* BASES DE DATOS :
*              PRESBASE.DBF
*  PBASICOS.DBF
*              PDETALLE.DBF
*              PUSER.DBF
*              PCOMPANY.DBF
*              PROVE.DBF
*              PVENDE.DBF
*              PCLIENTE.DBF
* NOTAS        :
*              SISTEMA CONTROLADOR DE PRESUPUESTOS
*              APLICACION PARA CLIPPER
#INCLUDE 'DBFNDX.CH'
MPAN = SAVESCREEN(0,0,24,79)

SET PROC TO PPRO
STARTCOLOR = SETCOLOR()
SET KEY 301 TO DOQUIT
SET KEY 274 TO DOSSHELL
SET KEY -5 TO MSTATUS()
SET KEY -4 TO MSTRUC()
DO PENTRADA
IF MNIVEL = 0
  CLOSE ALL
  RESTSCREEN(00,00,24,79,MDOS)
  @ 23,00 SAY ''
  QUIT
ENDI
LAPSO = MREVDATE-DATE()
*IF LAPSO
*  @ 20,05 CLEAR TO 23,70
*  @ 20,15 SAY 'NOTA:'
*  @ 21,20 SAY 'FAVOR DE LLAMAR AL 589-81-90 PARA SOLICITAR'
*  @ 22,20 SAY 'CAMBIO DE VERSION O ACTUALIZACION DE SISTEMA'
*  @ 23,20 SAY 'INDICAR REFERECIA VER.: '+MVERSION+STR(LAPSO)
*  ! DEL PRODUCTO.DBF
*  ! REN P.EXE PRODUCTO.DBF
*  INKEY(0)
*  CLEAR
*ENDI
*IF MREVDATEATE()
*  CLOSE ALL
*  QUIT
*ENDI
SET WRAP ON                  && MOVIMIENTO ROTATORIO EN MEUS
SET MESS TO 24 CENTER        && DESPLIEGA MENSAJES EN LINEA 24
CLEAR
PADS = 1                     && VARIABLE DE MEMORIA MENU PRINCIPAL
POPU = 1                     && VARIABLE DE MEMORIA MENU POPUP
KEYBOARD CHR(13)             && ACTIVAR MENU DESDE ENTRADA
DO PTEXTOS
DO WHIL .T.
  IF LASTKEY() = 27
    RESTSCREEN(01,00,23,79,MPFONDO)
  ENDI
  * PRESUPUESTOS BASICOS PEDIDOS REMISIONES FACTURAS CATALOGOS    TERMINAR
*
01234567890123456789012345678901234567890123456789012345678901234567890123456789
  *      1      2      3      4      5      6      7
  SET COLOR TO &C_CAJA
  @ 00,00
  @ 24,00
  @ 00,01 PROM 'PRESUPUESTOS' MESS 'CONTROL DE PRESUPUESTOS'
  @ 00,14 PROM 'BASICOS'      MESS 'CONTROL DE PRODUCTOS BASICOS O DE LINEA'
  @ 00,22 PROM 'PEDIDOS'      MESS 'CONTROL DE PEDIDOS'
  @ 00,30 PROM 'REMISIONES'   MESS 'CONTROL DE REMISIONES'
  @ 00,41 PROM 'FACTURAS'     MESS 'CONTROL DE FACTURA'
  @ 00,50 PROM 'O.COMPRAS'    MESS 'CONTROL DE ORDENES DE COMPRA'
  @ 00,60 PROM 'CATALOGOS'    MESS 'CATALOGOS ADICIONALES AL SISTEMA'
  @ 00,70 PROM 'TERMINAR'     MESS 'SALIDA A SISTEMA OPERATIVO'
  MENU TO PADS
  IF PADS#0
    DO SELE_PADS WITH PADS
  ELSE
    RESTSCREEN(01,00,23,79,MPFONDO)
  ENDI
  LOOP
ENDD


PROC SELE_PADS
PARA PAD
* ASIGNA FLECHAS PARA SIGIENTE MENU ISQUIERDA O DERECHA
SET KEY 19 TO FLECHA_IZQ
SET KEY  4 TO FLECHA_DER
RELEASE POPU                 && LIMPIA LA VARIABLE POPU
SET COLOR TO &C_TEXT

RESTSCREEN(01,00,23,79,MPFONDO)
SET COLOR TO &C_CAJA
DO CASE
  CASE PAD = 1
    @ 01,01 CLEAR TO 13,27
*   @ 01,01 TO 13,27
    @ 02,03 PROM 'MOVIMIENTOS '
    @ 03,03 PROM 'ORDENAR     '
    @ 04,03 PROM 'SUPRIMIR    '
    @ 05,02 SAY 'MMMMMMMMMMMMMMMMMMMMMMMMM'
    @ 06,03 PROM 'ENVIO A DRIVE      '+CHR(25)+''
    @ 07,03 PROM 'REGRESAR A DISCO DURO '+CHR(24)+''
    @ 08,02 SAY 'MMMMMM CONTENIDOS MMMMMMM'
    @ 09,05 PROM 'MOVIMIENTOS '
    @ 10,05 PROM 'ORDENAR     '
    @ 11,05 PROM 'SUPRIMIR    '
    @ 12,05 PROM 'RECONSTRUIR '
*     @ 13,26 PROM 'AJUSTAR (%)'
*     @ 14,26 PROM 'INCREMENTOS'
*     @ 15,26 PROM 'RESTABLECER'
    SOMBRA(1,1,13,27)
  CASE PAD = 2
```

```
      @ 01,14 CLEAR TO 09,29
*     @ 01,14 TO 09,29
      @ 02,15 PROM 'MOVIMIENTOS '
      @ 03,15 PROM 'ORDENAR     '
      @ 04,15 PROM 'SUPRIMIR    '
      @ 05,15 PROM 'INCREMENTOS '
      @ 06,15 PROM 'RESTABLECER '
      @ 07,15 SAY 'ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ'
      @ 08,15 PROM 'IMPRESIÓN   '
      SOMBRA(1,14,9,29)
    CASE PAD = 3
      @ 01,22 CLEAR TO 09,49
*     @ 01,22 TO 09,49
      @ 02,24 PROM 'MOVIMIENTOS '
      @ 03,24 PROM 'ORDENAR     '
      @ 04,24 PROM 'SUPRIMIR    '
'ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ' @ 05,23 SAY
      @ 06,24 PROM 'ENVIO A DRIVE     ' + CHR(25) + ''
      @ 07,24 PROM 'REGRESAR A DISCO DURO ' + CHR(24) + '
      @ 08,24 PROM 'RECONSTRUIR       '
      SOMBRA(1,22,9,49)
    CASE PAD = 4
      @ 01,30 CLEAR TO 09,57
*     @ 01,30 TO 09,57
      @ 02,32 PROM 'MOVIMIENTOS '
      @ 03,32 PROM 'ORDENAR     '
      @ 04,32 PROM 'SUPRIMIR    '
      @ 05,31 SAY 'ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ'
      @ 06,32 PROM 'ENVIO A DRIVE     ' + CHR(25) + ''
      @ 07,32 PROM 'REGRESAR A DISCO DURO ' + CHR(24) + '
      @ 08,32 PROM 'RECONSTRUIR       '
      SOMBRA(1,30,9,57)
    CASE PAD = 5
      @ 01,41 CLEAR TO 09,65
*     @ 01,41 TO 09,65
      @ 02,43 PROM 'MOVIMIENTOS '
      @ 03,43 PROM 'ORDENAR     '
      @ 04,43 PROM 'SUPRIMIR    '
      @ 05,42 SAY 'ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ'
      @ 06,43 PROM 'ENVIO A DRIVE     ' + CHR(25) + ''
      @ 07,43 PROM 'REGRESAR A DISCO DURO ' + CHR(24) + '
      @ 08,43 PROM 'RECONSTRUIR       '
      SOMBRA(1,41,9,65)
    CASE PAD = 6
      @ 01,50 CLEAR TO 09,77
*     @ 01,50 TO 09,77
      @ 02,52 PROM 'MOVIMIENTOS '
      @ 03,52 PROM 'ORDENAR     '
      @ 04,52 PROM 'SUPRIMIR    '
      @ 05,51 SAY 'ÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ'
      @ 06,52 PROM 'ENVIO A DRIVE     ' + CHR(25) + ''
      @ 07,52 PROM 'REGRESAR A DISCO DURO ' + CHR(24) + '
      @ 08,52 PROM 'RECONSTRUIR       '
      SOMBRA(1,50,9,77)
    CASE PAD = 7
      @ 01,60 CLEAR TO 09,74
*     @ 01,60 TO 09,74
      @ 02,62 PROM 'VENDEDORES '
      @ 03,62 PROM 'CLIENTES   '
      @ 04,62 PROM 'PROVEEDORES'
      @ 05,62 PROM 'USUARIOS   '
      @ 06,62 PROM 'ACCESOS    '
      @ 07,62 PROM 'COMPAÑIAS  '
      SOMBRA(1,60,8,74)
    CASE PAD = 8
      @ 01,68 CLEAR TO 05,79
      SOMBRA(1,68,5,79)
*     @ 01,68 TO 05,79
      @ 02,70 PROM 'CONTINUA'
      @ 03,69 SAY 'ÍÍÍÍÍÍÍÍÍÍ'
      @ 04,70 PROM 'TERMINAR'
  ENDC

  MENU TO POPU
  * ASIGNA FLECAHAS PARA USO NORMAL
  SET KEY 19 TO
  SET KEY  4 TO
  DO CASE
    CASE PAD = 1
      DO CASE
        CASE POPU = 1
          DO PRESMOVE
          KEYBOARD CHR(13)
        CASE POPU = 2
          SELE 1
          @ 24,00
                @ 24,00 SAY 'ESPERA UN MOMEMTO
RECONTRUYENDO INDICES...'
          REIN
          ?? CHR(7)
          ?? CHR(7)
          KEYBOARD CHR(13)
        CASE POPU = 3
          SELE 1
          @ 24,00
                @ 24,00 SAY 'ESPERA UN MOMEMTO ELIMINANDO
REGISTROS MARCADOS...'
          PACK
          ?? CHR(7)
          ?? CHR(7)
          KEYBOARD CHR(13)
        CASE POPU = 4
          DO PEAJPRES
          KEYBOARD CHR(13)
        CASE POPU = 5
          DO PSUBPRES
          KEYBOARD CHR(13)
        CASE POPU = 6
          SELE 3
          MIRELA = '00-00000'
          DO PDETMOVE
          KEYBOARD CHR(13)
        CASE POPU = 7
          SELE 3
          @ 24,00
                @ 24,00 SAY 'ESPERA UN MOMEMTO
RECONTRUYENDO INDICES...'
          REIN
          ?? CHR(7)
          ?? CHR(7)
          KEYBOARD CHR(13)
        CASE POPU = 8
          SELE 3
          @ 24,00
                @ 24,00 SAY 'ESPERA UN MOMEMTO ELIMINANDO
REGISTROS MARCADOS...'
          PACK
          ?? CHR(7)
```

```
            ??? CHR(7)
            KEYBOARD CHR(13)
          CASE I-OPU = 8
            DO PRESPURG
            KEYBOARD CHR(13)
        ENDC
      CASE PAD = 2
        DO CASE
          CASE POPU = 1
            SELE 2
            &REL = 0
            DO PBASMOVE
            KEYBOARD CHR(13)
          CASE POPU = 2
            SELE 2
            @ 24,00
                  @ 24,00 SAY "ESPERA UN MOMEMTO
RECONTRUYENDO INDICES..."
            REIN
            ??? CHR(7)
            ??? CHR(7)
            KEYBOARD CHR(13)
          CASE POPU = 3
            SELE 2
            @ 24,00
              @ 24,00 SAY "ESPERA UN MOMEMTO ELIMINANDO
REGISTROS MARCADOS..."
            PACK
            ??? CHR(7)
            ??? CHR(7)
            KEYBOARD CHR(13)
          CASE POPU = 4
            DO PINCREME
            KEYBOARD CHR(13)
          CASE POPU = 5
            SELE 2
            GO TOP
            SET COLOR TO &C_CAJA
            @ 24,00
            @ 24,00 SAY RECC()
            DO WHIL .NOT. EOF()
              @ 24,20 SAY RECN()
              REPLACE OPERA WITH "O"
              SKIP
            ENDD
            KEYBOARD CHR(13)
          CASE POPU = 6
            DO PIMPBASI
            KEYBOARD CHR(13)
        ENDC
      CASE PAD = 3
        DO CASE
          CASE POPU = 1
            DO PEDIMOVE
            KEYBOARD CHR(13)
          CASE POPU = 2
            SELE 4
            USE PEDIDOS INDEX PEDIDOS VIA "DBFNDX"
            @ 24,00
                  @ 24,00 SAY "ESPERA UN MOMEMTO
RECONTRUYENDO INDICES..."
            REIN
            USE PUSER
            KEYBOARD CHR(13)
          CASE POPU = 3
            SELE 4
            USE PEDIDOS INDEX PEDIDOS VIA "DBFNDX"
            @ 24,00
              @ 24,00 SAY "ESPERA UN MOMEMTO ELIMINANDO
REGISTROS MARCADOS..."
            PACK
            USE PUSER
            KEYBOARD CHR(13)
          CASE POPU = 4
            DO PBAJPEDI
            KEYBOARD CHR(13)
          CASE POPU = 5
            DO PSUBPEDI
            KEYBOARD CHR(13)
          CASE POPU = 6
            DO PEDIPURG
            KEYBOARD CHR(13)
        ENDC
        SELE 3
        USE PDETALLE INDEX PDETALLE VIA "DBFNDX"
        SELE 4
        USE PUSER
      CASE PAD = 4
        DO CASE
          CASE POPU = 1
            DO PREMMOVE
            KEYBOARD CHR(13)
          CASE POPU = 2
            SELE 4
            USE PREMISIO INDEX PREMISIO VIA "DBFNDX"
            @ 24,00
                  @ 24,00 SAY "ESPERA UN MOMEMTO
RECONTRUYENDO INDICES..."
            REIN
            USE PUSER
            KEYBOARD CHR(13)
          CASE POPU = 3
            SELE 4
            USE PREMISIO INDEX PREMISIO VIA "DBFNDX"
            @ 24,00
              @ 24,00 SAY "ESPERA UN MOMEMTO ELIMINANDO
REGISTROS MARCADOS..."
            PACK
            USE PUSER
            KEYBOARD CHR(13)
          CASE POPU = 4
            DO PBAJREMI
            KEYBOARD CHR(13)
          CASE POPU = 5
            DO PSUBREMI
            KEYBOARD CHR(13)
          CASE POPU = 6
            DO PREMPURG
            KEYBOARD CHR(13)
        ENDC
        SELE 3
        USE PDETALLE INDEX PDETALLE VIA "DBFNDX"
        SELE 4
        USE PUSER
      CASE PAD = 5
        DO CASE
          CASE POPU = 1
            DO PFACMOVE
            KEYBOARD CHR(13)
          CASE POPU = 2
            SELE 4
            USE PFACTURA INDEX PFACTURA VIA "DBFNDX"
```

```
            @ 24,00
                        @ 24,00 SAY "ESPERA UN MOMENTO
RECONTRUYENDO INDICES..."
            REIN
            USE PUSER
            KEYBOARD CHR(13)
         CASE POPU = 3
            SELE 4
            USE PFACTURA INDEX PFACTURA VIA "DBFNDX"
            @ 24,00
                  @ 24,00 SAY "ESPERA UN MOMENTO ELIMINANDO
REGISTROS MARCADOS..."
            PACK
            USE PUSER
            KEYBOARD CHR(13)
         CASE POPU = 4
            DO PBAJFACT
            KEYBOARD CHR(13)
         CASE POPU = 5
            DO PSUBFACT
            KEYBOARD CHR(13)
         CASE POPU = 6
            DO PFACPURG
            KEYBOARD CHR(13)
      ENDC
      SELE 3
      USE PDETALLE INDEX PDETALLE VIA "DBFNDX"
      SELE 4
      USE PUSER
   CASE PAD = 6
      DO CASE
         CASE POPU = 1
            DO PORDMOVE
            KEYBOARD CHR(13)
         CASE POPU = 2
            SELE 4
            USE PORDCOMP INDEX PORDCOMP VIA "DBFNDX"
            @ 24,00
                  @ 24,00 SAY "ESPERA UN MOMENTO
RECONTRUYENDO INDICES..."
            REIN
            USE PUSER
            KEYBOARD CHR(13)
         CASE POPU = 3
            SELE 4
            USE PORDCOMP INDEX PORDCOMP VIA "DBFNDX"
            @ 24,00
                  @ 24,00 SAY "ESPERA UN MOMENTO ELIMINANDO
REGISTROS MARCADOS..."
            PACK
            USE PUSER
            KEYBOARD CHR(13)
         CASE POPU = 4
            DO PBAJORDC
            KEYBOARD CHR(13)
         CASE POPU = 5
            DO PSUBORDC
            KEYBOARD CHR(13)
         CASE POPU = 6
            DO PORDPURG
            KEYBOARD CHR(13)
      ENDC
      SELE 3
      USE PDETALLE INDEX PDETALLE VIA "DBFNDX"
      SELE 4
      USE PUSER
```

```
   CASE PAD = 7
      DO CASE
         CASE POPU = 1
            SELE 1
            DO PVENMOVE
            KEYBOARD CHR(13)
         CASE POPU = 2
            SELE 1
            DO PCLIMOVE
            KEYBOARD CHR(13)
         CASE POPU = 3
            SELE 1
            DO PPROVMOVE
            KEYBOARD CHR(13)
         CASE POPU = 4
            SELE 4
            DO PUSEMOVE
            KEYBOARD CHR(13)
         CASE POPU = 5
            SELE 4
            DO PACCESOS
            KEYBOARD CHR(13)
         CASE POPU = 8
            DO PCOMMOVE
            DO PTEXTOS
            KEYBOARD CHR(13)
      ENDC
   CASE PAD = 8
      DO CASE
         CASE POPU = 1
            KEYBOARD CHR(13)
         CASE POPU = 2
            @ 23,00 SAY ""
            CLOSE ALL
            RESTSCREEN(0,0,24,79,MPAN)
            DO SALIDA
            QUIT
      ENDC
ENDC
RETURN



PROC FLECHA_IZQ
PARA PN, PL, RV
* RETURN, UP ARROW, RETURN
KEYBOARD CHR(27) + CHR(5) + CHR(13)
RETURN


PROC FLECHA_DER
PARA PN, PL, RV
* RETURN, RIGHT ARROW, RETURN
KEYBOARD CHR(27) + CHR(4) + CHR(13)
RETURN
```

PENTRADA.PRG

```
*_____*
* PROGRAMA    : PENTRADA.PRG          *
* REALIZADO POR ARTURO CROTTE FRANCO. *
* BASES DE DATOS :                    *
*          PRESBASE.DBF               *
*          POETALLE.DBF               *
*          PUSER.DBF                  *
* NOTAS :                             *
*     ACCESO A PERSONAL AUTORIZADO    *
*_____*

SET TALK OFF
SET STAT OFF
SET SCOR OFF
SET DATE ITALIAN                && ACTIVACION DE FECHA EN DD-MM-AA

PUBL MNIVEL, MCIA, MTITULAR, MUSUARIO, REA, REI, REF, MNUMERO, MREVDATE

PUBL MRELA, MPRECIO, MREFE, MNUMERO, MFECHA, MES, MLETRA, MVERSION

MREVDATE = CTOD('31-12-93')
MVERSION = '3.0  DIC.31/1993'
STORE 0 TO MNIVEL, CHANCE, MCIA, REA, REI, REF, MPRECIO, MNUMERO
STORE SPACE(60) TO MTITULAR
STORE SPACE(20) TO MUSUARIO
STORE SPACE(8) TO MRELA,MREFE
PRIVATE RA,RI,RF
STORE 0 TO RA,RI,RF
STORE DATE() TO MFECHA
PUBL C_TEXT,C_ENTRA,C_CAJA,C_VENTANA,C_RESALTA
STORE SPACE(20) TO MES
STORE SPACE(50) TO MCLIENTE,MPROVEEDOR
STORE 0 TO REA,REI,REF,MNIVEL,CHANCE,MCMP,MRA,MNUMERO
STORE SPACE(12) TO MIDE,MLETRA
STORE DATE() TO MFECHA

IF ISCOLOR()
  C_CAJA  = "BG/B,BG/N ,B+,,W+/B"
  C_TEXT  = "N+/W,BG+/N,B+,,W+/B"
  C_ENTRA  = "N/BG,BG+/N,B+,,N/BG"
  C_VENTANA = "BG/N,B,N ,B+,,B+/N"
  C_RESALTA = "W+/B,BG+/N,B+,,W+/B"
ELSE
  C_CAJA  = "N/W ,W,N ,B+,,N/W"
  C_TEXT  = "N+/W,W+/N,B+,,W/N"
  C_ENTRA  = "N+/W,W+/N,B+,,W/N"
  C_VENTANA = "N/W ,W,N ,B+,,W+/N"
  C_RESALTA = "W+/N,W+/N,B+,,W+/N"
ENDIF
SET COLOR TO &C_TEXT
CLEAR
DO WHILE .T.
  LEVEL = 0
  SET COLOR TO &C_RESALTA
  DO CRECE WITH 3,19,18,60,"SR+/B",.T.
  @ 4, 20 SAY '      PRESUPUESTOS      '
  @ 14, 20  SAY 'VER.' + MVERSION
  @ 15, 34  SAY 'CLIPPER 5'
  SET COLO TO &C_RESALTA
  @ 2, 19 TO 18, 60
  @ 6, 20  SAY 'CREADO POR : ARTURO CROTTE FRANCO.'
  @ 7, 20  SAY '           TABLAS DE CONSULTA'
  @ 8, 20  SAY '           ALGORITMO MOVE 4'
  @ 9, 20  SAY '           5-09-01-90'
  @ 17, 22 SAY 'DAME TU CLAVE DE ACCESO : '
  SET COLO TO &C_ENTRA
  SELE 1
  USE PUSER
  @ 17, 48 SAY '0000'
  @ 17, 48 SAY ''
  X = 47
  A = 1
  MCLAVE = ''
  DO WHIL A
    DO WHIL .T.
      I = 0
      DO WHIL I = 0
        I = INKEY()
      ENDO
      IF I # 13
        EXIT
      ENDI
    ENDO
    MCLAVE = MCLAVE + CHR(I)
    A = A + 1
    X = X + 1
    @ 17, X SAY CHR(S)
  ENDO
  SET COLOR TO W/B,W+/N,W
  GO TOP
  MCLAVE = UPPER(MCLAVE)
  MENC = ''
  FOR ME = 1 TO LEN(MCLAVE)
    MENC = MENC + CHR(ASC(SUBSTR(MCLAVE,ME,1)) + 100)
  NEXT
  MCLAVE = MENC
  LOCATE FOR CLAVE = MCLAVE
  IF FOUND()
    EXIT
  ELSE
    CHANCE = CHANCE + 1
    ?? CHR(7) + CHR(7)
    @ 22,18 SAY 'CLAVE INCORRECTA... PRESIONA ALGUNA TECLA.'
    INKEY(0)
    SET COLO TO &C_CAJA
    IF CHANCE = 3
      @ 22,18 SAY 'LO SIENTO, USTED NO TIENE ACCESO AL SISTEMA.'
      INKEY(5)
      CLOSE ALL
      RESTSCREEN(00,00,24,79,MPAN)
      @ 23,00 SAY ''
      QUIT
    ENDIF
  ENDDO

  MUSUARIO = NOMBRE
  MCIA = CIA
  MNIVEL = NIVEL
  USE PACCESOS
  APPE BLAN
  REPL NOMBRE WITH MUSUARIO
  REPL FECHA WITH DATE()
```

```
REPL HORA  WITH TIME()
CLOSE DATABASE
CLOSE INDEX
SELE 1
USE PRESBASE INDEX PRESBASE VIA "DBFNDX"
SELE 2
USE PBASICOS INDEX PBASICOS VIA "DBFNDX"
SELE 3
USE PDETALLE INDEX PDETALLE VIA "DBFNDX"
SELE 4
USE PUSER
SELE 5
USE PCOMPANY INDEX PCOMPANY VIA "DBFNDX"
SEEK MCIA
IF FOUND()
  MTITULAR = TITULAR
ENDIF
RETU
```

PPRO.PRG (VER APENDICE A)

TODO LO REFERNTE A ESTE PROGRAMA EN EL APENDICE A.

PJUSTIFI.PRG

```
**********************************************
**
 * PROGRAMA PARA JUSTIFICAR CAMPOS MEMO A MARGEN VARIABLE
 * ROBERTO BARRON CORTES
 **********************************************
PARA TEXTO,LONG
TEXTO = LTRIM(RTRIM(TEXTO))
JA = LEN(TEXTO)
MCINT = 0                    && CONTADOR DE INTENTOS DE LINEA JUSTIFICADA
IF JA = LONG .OR. JA = 0
  RETU
ENDI
IF SUBS(TEXTO,JA,1)$'.' .OR. SUBS(TEXTO,JA,1) = CHR(13)
  RETU
ENDI
BLAN = 0
JX = 1
  DO WHIL JX<A
  T = INKEY()
  IF T = 27
    EXIT
  ENDI
  JC = SUBS(TEXTO,JX,1)
  IF JC = ' '
    BLAN = BLAN + 1
    POS = 'POS' + LTRI(STR(BLAN))
    &POS = JX
  ENDI
  JX = JX + 1
ENDD
IF T = 27
  RETU
ENDI
DO WHIL JA > N
 * ADICION DE UN CARACTER EN CASO DE NO JUSTIFICAR ***
  MCINT = MCINT + 1
  IF MCINT.LONG
    TEXTO = TEXTO + ' ~'
    RETU
  ENDI
  **********************************************
  T = INKEY()
  IF T = 27
    @ 24,00 SAY MCINT
    INKEY(0)
    EXIT
  ENDI
  JX = 1
  DO WHIL JA < N
    POS = 'POS' + LTRI(STR(JX))
    TEXTOL = LEFT(TEXTO,&POS)
    TEXTOR = RIGHT(TEXTO,LEN(TEXTO)-&POS)
    TEXTO = TEXTOL + ' ' + TEXTOR
    JA = LEN(TEXTO)
    IF JA = LONG
      EXIT
    ENDI
    JX = JX + 1
    JY = JX
    T = INKEY()
    IF T = 27
      EXIT
    ENDI
    DO WHIL J > N
      POS = 'POS' + LTRI(STR(JY))
      &POS = &POS + 1
      JY = JY + 1
      T = INKEY()
      IF T = 27
        EXIT
      ENDI
    ENDD
  ENDD
ENDD
```

PRESMOVE.PRG

```
*************************************************************
 * PROGRAMA.............: PRESMOVE.PRG
 * NOTAS................: MOVIMIENTO ALTAS Y BAJAS CAMBIOS DE PRESUPUESTOS
 *
 * BASE DE DATOS........: PRESBASE.DBF
 * INDEX ...............: PRESBASE.NDX (FECHA)
 * SELECT ..............: 1
 * FECHA ...............: 03-MARZO-1991
 * VERSION .............: 1.0
 * REALIZADO POR       : ARTURO CROTTE FRANCO
ARTURO CROTTE FRANCO
 *************************************************************
MP1 = SAVESCREEN(00,00,24,79)
SAVESCREEN(00,00,24,79,MP1)
SET ESCAPE ON
SET EXACT OFF
SET DELE OFF
SET BELL OFF
STORE 0 TO REA,REI,REF,T,RPF,RPUPA
STORE SPACE(8) TO MREFE,BUSC,MRELA
SELE 1

SET COLOR TO &C_RESALTA
@ 23,00
  @ 23,00 SAY " ESPERE UN MOMENTO ACTUALIZANDO CONSULTA DE CUENTES."
*****************************************
*** TABLA DE CUENTES        ***
*****************************************
SELE 1
USE PCUENTE INDEX PCUENTE VIA "DBFNDX"
TOTAL = RECC()
RELEASE MCUENTES
DECLARE MCUENTES[TOTAL]
MC = 1
GO TOP
DO WHILE .NOT.EOF()
  MCUENTES[MC] = NOMBRE
  SKIP
  MC = MC + 1
ENDDO
USE
*****************************************
SELE 3
USE PDETALLE INDEX PDETALLE VIA "DBFNDX"
SELE 1
USE PRESBASE INDEX PRESBASE VIA "DBFNDX"
******* FORMACION DE PANTALLA DE VISUALIZACION*****
SET COLOR TO N + /W,BG + /N,B + ,N/BG
CLEAR
SET COLOR TO B + /B
@ 0, 0 SAY REPLICATE("\",80)
SET COLOR TO W + /B
@ 0,33 SAY " PRESBASE.DBF "
SET COLOR TO N + /W
@ 1,48 TO 7,78 DOUBLE
@ 2,13 SAY "DISE%O PROFESIONAL DE EQUIPOS"
@ 2,54 SAY "P R E S U P U E S T O"
@ 3,16 SAY "Y COCINAS, S.A. DE C.V."
@ 4,51 SAY "REF"
@ 5,12 SAY "N#MERO DE PRESUPUESTO :"
@ 6,12 SAY "TOTAL DE PRESUPUESTOS :"
@ 5,51 SAY "FECHA"
@ 6,51 SAY "AGENTE"
@ 8,2 TO 20,78 DOUBLE
@ 9,5 SAY "NOMBRE....."
@ 10,5 SAY "ATENCION..."
@ 11,5 SAY "DIRECCION..."
@ 12,5 SAY "TELEFONOS.."
@ 13,5 SAY "PLAZO DE"
@ 14,5 SAY "ENTREGA...."
@ 15,5 SAY "FORMA DE..."
@ 16,5 SAY "P A G O...."
@ 18,5 SAY "OBSERVACIONES:"
*************************************************************
GO TOP
REI = RECNO()
GO BOTT
REF = RECNO()
REA = RECNO()
*************** LOOP - PRINCIPAL **********************
DO WHILE .T.
  IF RECC() # 0
    GO REA
  ENDI
  ******* VISUSLIZACION DE L REGISTRO ACTUAL ********
  SET COLOR TO &C_ENTRA
  @ 4,62 SAY REFE PICTURE "99-99999"
  @ 5,38 SAY RECNO() PICTURE "999,999"
  @ 6,38 SAY RECCOUNT() PICTURE "999,999"
  @ 5,62 SAY FECHA
  @ 8,62 SAY LEFT(AGENTE,15)
  @ 9,16 SAY NOMBRE
  @ 10,16 SAY ATENCION
  @ 11,16 SAY LEFT(DIRECCION,50)
  @ 12,16 SAY TELEFONOS
  @ 14,16 SAY PLAZO
  @ 15,16 SAY LEFT(FORMA1,60)
  @ 16,16 SAY LEFT(FORMA2,60)
  @ 17,16 SAY LEFT(FORMA3,60)
  @ 19, 7 SAY LEFT(NOTAS,69)
  ****** AVISO DEL ESTADO ACUAL DEL REGISTRO ******
  SET COLOR TO W/B
  @ 24,00
  @ 24,01 SAY "REG. #"
          @ 24,08 SAY ALLTRIM(STR(RECN())) + " DE
" + ALLTRIM(STR(RECC()))
  DO CASE
    CASE RECC() = 1
      @ 24,23 SAY " 1 SOLO REGISTRO   "
    CASE RECC() = 0
      @ 24,23 SAY " ARCHIVO SIN DATOS  "
    CASE REA = REF
      @ 24,23 SAY " FIN DE ARCHIVO    "
    CASE REA = REI
      @ 24,23 SAY " PRINCIPIO DE ARCHIVO"
  ENDC
  IF DELETED()
    SET COLOR TO W + */B
    @ 24,44 SAY " MARCADO "
  ENDIF
  SET COLOR TO N/BG
  ********** SELECCION DE LA TECLA DE CONTROL *******
  SET COLOR TO &C_CAJA
  @ 23,00
```

```
    @ 23,00 SAY "ALTA BUSCA CAMBIO HOJA MARCA
PRODUCTOS EXTRAS IMPRIMIR SUPRIMIR"
0123456789012345678901234567890123456789012345678901
23456789012345678901234
*          1    2    3    4    5    6    7
   SET COLOR TO &C_RESALTA
   @ 23,00 SAY "A"
   @ 23,05 SAY "B"
   @ 23,11 SAY "C"
   @ 23,18 SAY "H"
   @ 23,23 SAY "M"
   @ 23,29 SAY "P"
   @ 23,36 SAY "E"
   @ 23,46 SAY T
   @ 23,55 SAY "S"
   @ 24,55 SAY CHR(25) + CHR(24) + " PGUP PGDN HOME END
ESC"
   @ 23,79 SAY "
   ******** ACION DE LA TECLA PRESIONADA *********
   T = 0
   DO WHILE T = 0
     T = INKEYTRAP()
   ENDDO
   SET COLOR TO &C_CAJA
   @ 23,00
   DO CASE
     CASE T = 1    &&    TECLA    HOME      1 = DB3    26
= DB4
       GO TOP
       REA = RECNO()
     CASE T = 8    &&    TECLA    END       6 = DB3    2 = DB4
       GO BOTTOM
       REA = RECNO()
     CASE T = 18   &&    TECLA    PGUP
       S = 0
       DO WHILE .NOT. BOF()
         SKIP -1
         REA = RECNO()
         S = S + 1
         IF S = 10
           EXIT
         ENDIF
       ENDDO
     CASE T = 3    &&    TECLA    PGDN
       S = 0
       DO WHILE .NOT. EOF()
         SKIP
         REA = RECNO()
         S = S + 1
         IF S = 10
           EXIT
         ENDIF
       ENDDO
       IF EOF()
         SKIP -1
         REA = RECNO()
       ENDI
     CASE T = 24   &&    TECLA    FLECHA ABAJO
       IF REA = REF
         LOOP
       ELSE
         SKIP 1
         REA = RECNO()
       ENDIF
     CASE T = 5    &&    TECLA    FLECHA ARRIBA
       IF REA = REI
         LOOP
       ELSE
         SKIP -1
         REA = RECNO()
       ENDIF
     CASE T = 27   &&    TECLA    ESC
       EXIT
     CASE CHR(T)$"BB"
       ********** BUSQUEDA SEEK FIND LOCATE *******
       MREFE  =  REFE
       @ 4,62 GET MREFE PICTURE "99-99999"
       READ
       BUSC = MREFE
       BUSC = TRIM(MREFE)
       FIND &BUSC
       IF FOUND()
         REA = RECNO()
       ENDI
     CASE CHR(T)$"AACC"
       ********CAPTURA DE DATOS DEL REGISTO ACUTUAL **
       SET COLOR TO &C_CAJA
       @ 23,00 CLEAR
       @ 23,10 SAY " EDITANTO DATOS DEL REGISTRO "
       SET COLOR TO &C_ENTRA
       IF CHR(T)$"AA"
         GO BOTT
           MREFE = LEFT(REFE,2) + "-" + RIGHT("00000" + FITRIM
(LTRIM(STR(VAL(RIGHT(REFE,5)) + 1))),5)
         @ 5,36 SAY RECNO() + 1 PICTURE "999,999"
         @ 4,62 GET MREFE PICTURE "99-99999"
         READ
         IF LASTKEY() = 27
           LOOP
         ENDI
         BUSC = MREFE
         BUSC = TRIM(BUSC)
         FIND &BUSC
         IF FOUND()
           OP = " "
           SET COLOR TO &C_CAJA
           @ 23,00 CLEAR
           @ 23,30 SAY "YA EXISTE"
           @ 24,30 SAY "CONTINUAR /N..."
           @ 24,50 GET OP PICTURE "!" VALID OP$"SN"
           READ
           IF OP = "N"
             LOOP
           ENDIF
         ENDI
         IF LASTKEY() = 27
           EXIT
         ENDI
         APPEN BLANK
         REPLACE REFE WITH MREFE
         REPL FECHA WITH DATE()
         REF = RECNO()
         @ 23,0 CLEAR
             @ 23,10 SAY " AGREGANDO UN REGISTRO AL
ARCHIVO.              (ALTA) "
       ENDI
       REA = RECNO()
       SET COLOR TO &C_ENTRA
       VIEJO = REFE
       MNOMBRE = NOMBRE
       IF CHR(T) = "C" .OR. CHR(T) = "C"
         @ 4,62 GET REFE PICTURE "99-99999"
```

```
      ENDI
      SET KEY -8 TO CON_CLIE
      SET COLOR TO &C_RESALTA
      @ 23,00
          @ 23,10 SAY ' EDITANTO DATOS DEL REGISTRO
= CUENTES'
      SET COLOR TO &C_ENTRA
      @ 5,36 SAY RECNO() PICTURE '999,999'
      @ 5,62 GET FECHA
      @ 6,62 GET AGENTE PICTURE '@S15'
      @ 9,16 GET MNOMBRE PICT '@!'
      READ
      SET KEY -8 TO
      MNOMBRE = ALLTRIM(MNOMBRE)
      IF .NOT. EMPTY(MNOMBRE)
        SELE 2
        USE PCLIENTE INDEX PCLIENTE VIA 'DBFNDX'
        FIND &MNOMBRE
        IF FOUND()
          MNOMBRE = NOMBRE
          MATE = ATENCION
          MDIR = DIRECCION
          MTEL = TELEFONOS
          MPLA = PLAZO
          MFO1 = FORMA1
          MFO2 = FORMA2
          MFO3 = FORMA3
          MNOT = NOTAS
          USE PBASICOS INDEX PBASICOS VIA 'DBFNDX'
          SELE 1
          REPLACE NOMBRE  WITH MNOMBRE
          REPLACE ATENCION WITH MATE
          REPLACE DIRECCION WITH MDIR
          REPLACE TELEFONOS WITH MTEL
          REPLACE PLAZO   WITH MPLA
          REPLACE FORMA1  WITH MFO1
          REPLACE FORMA2  WITH MFO2
          REPLACE FORMA3  WITH MFO3
          REPLACE NOTAS   WITH MNOT
        ENDI
        SELE 2
        USE PBASICOS INDEX PBASICOS VIA 'DBFNDX'
        SELE 1
      ENDI
      SET COLOR TO &C_CAJA
      @ 23,00
      @ 23,10 SAY ' EDITANDO DATOS DEL REGISTRO '
      SET COLOR TO &C_ENTRA
      @ 10,16 GET ATENCION PICTURE '@!'
      @ 11,16 GET DIRECCION PICTURE '@!'
          @ 12,16 GET TELEFONOS PICTURE
'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
      @ 14,16 GET PLAZO PICTURE '@!'
      @ 15,16 GET FORMA1 PICTURE '@!'
      @ 16,16 GET FORMA2 PICTURE '@!'
      @ 17,16 GET FORMA3 PICTURE '@!'
      @ 19, 7 GET NOTAS PICTURE '@!'
      READ
      SET COLOR TO &C_CAJA
      IF REFEVIEJO .AND. CHR(T)$'CC'
        NUEVO = REFE
        SELE 3
        @ 23,00 CLEAR
        @ 23,00 SAY 'ESPERA POR UN MOMENTO...'
        CLOSE INDEX
        GO TOP
        DO WHIL .NOT. EOF()
          @ 23,40 SAY REFE
          IF REFE = VIEJO
            @ 23,60 SAY REFE
            REPLACE REFE WITH NUEVO
            @ 24,60 SAY REFE
          ENDIF
          SKIP
        ENDO
        @ 23,00 CLEAR
            @ 23,00 SAY 'ESPERA POR UN MOMENTO...
RECONTRUYENDO INDICES ...'
        USE PDETALLE INDEX PDETALLE VIA 'DBFNDX'
        REIN
        SELE 1
      ENDIF
      GO TOP
      REI = RECNO()
      GO BOTT
      REF = RECNO()
      GO REA
      LOOP
    CASE CHR(T)$'MM'
      ****** SECCION DELETE Y RECALL **********
      @ 23,00 CLEAR
      @ 23,02 SAY 'ESPERA UN MOMENTO...'
      IF DELETED()
        RECALL
        TEMP2 = REFE
        SELE 3
        FIND &TEMP2
        DO WHIL REFE = TEMP2 .AND. .NOT. EOF()
          IF REFE = TEMP2
            @ 24,00 SAY RECN() PICTURE '999999'
            @ 24,10 SAY REFE
            RECALL
            @ 24,20 SAY REFE
          ENDIF
          SKIP
        ENDO
        SELE 1
      ELSE
        DELETE
        TEMP2 = REFE
        SELE 3
        FIND &TEMP2
        DO WHIL REFE = TEMP2 .AND. .NOT. EOF()
          IF REFE = TEMP2
            @ 24,00 SAY RECN() PICTURE '999999'
            @ 24,10 SAY REFE
            DELETE
            @ 24,20 SAY REFE
          ENDIF
          SKIP
        ENDO
        SELE 1
      ENDIF
    CASE CHR(T)$'EE'
      MP2 = SAVESCREEN(06,10,19,71)
      SET COLOR TO &C_CAJA
      @ 6,10 CLEAR TO 18,70
      @ 6,10 TO 18,70
          @ 07,15 SAY ' ( % ) DESCUENTOS OTORGADOS AL
PRESUPUESTO'
      @ 08,15 SAY 'MARCA NACIONAL    :'
      @ 09,15 SAY 'FABRICACI'N ESPECIAL :'
```

```
        @ 10,15 SAY "IMPORTACI'N      :"
        @ 11,15 SAY "DESCUENTO POR FACTO :"
        @ 12,15 SAY "                 :"
      @ 13,19 SAY " CARGOS ADICIONALES AL PRESUPUESTO
        @ 14,15 SAY "FLETE            :"
        @ 15,15 SAY "EXTRACCI'N       :"
        @ 16,15 SAY "INSTALACI'N      :"
        @ 17,15 SAY "COLOCACI'N       :"
        @ 10,55 SAY "TIPO DE CAMBIO"
        @ 16,57 SAY "(%) DE I.V.A."
        SOMBRA(8,10,18,70)
        SET COLOR TO &C_ENTRA
        @  8,40 GET DESCMN   PICTURE "999.99"
        @  9,40 GET DESCFE   PICTURE "999.99"
        @ 10,40 GET DESCIM   PICTURE "999.99"
        @ 11,40 GET FACTO    PICTURE "999.99"
        @ 12,15 GET DESFINTEX PICTURE "@!"
        @ 12,40 GET DESFINIMP PICTURE "999.99"
        @ 14,40 GET FLETE    PICTURE "999,999,999.99"
        @ 15,40 GET EXTRAC   PICTURE "999,999,999.99"
        @ 16,40 GET INSTAL   PICTURE "999,999,999.99"
        @ 17,40 GET COLOCA   PICTURE "999,999,999.99"
        @ 11,57 GET T_CAMBIO PICTURE "999.99"
        @ 17,57 GET PIVA     PICTURE "99.9"
        READ
        RESTSCREEN(08,10,18,71,MP2)
      CASE CHR(T)$"HH"
        ******** VISUALIZACION EN FORMA TABULAR *********
        MP2 = SAVESCREEN(01,00,24,79)
        SAVESCREEN(01,00,24,79,MP2)
        SET COLOR TO &C_CAJA
        @ 24,00
* + CHR(205) + CHR(205) + CHR(27) + CHR(24) + " "
                                    @ 24,00 SAY " USE
        SET COLOR TO &C_TEXT
        DBEDIT(1,0,23,79)
        RESTSCREEN(01,00,24,79,MP2)
        RELEASE MP2
        REA = RECNO()
        LOOP
      CASE CHR(T)$"PP"
        RPA = REA
        RPI = REI
        RPF = REF
        PUBL MRELA
        MRELA = TRIM(REFE)
        SELE 3
        DO PDETMOVE
        T = 0
        SELE 1
        REA = RPA
        REI = RPI
        REF = RPF
      CASE CHR(T)$"II"
        RPA = REA
        RPI = REI
        RPF = REF
        MRELA = TRIM(REFE)
        SELE 3
        DO PIMPPRES
        SELE 1
        REA = RPA
        REI = RPI
        REF = RPF
      CASE CHR(T)$"SS"
        SET COLOR TO &C_RESALTA
        @ 24,00
        @ 24,00 SAY " ELIMINANDO REGISTROS MARCADOS:"
        SELE 3
        PACK
        SELE 1
        PACK
        GO TOP
        REI = RECNO()
        GO BOTT
        REA = RECNO()
        REF = RECNO()
      OTHER
        T = 0
        LOOP
    ENDC
  ENDD
  RESTSCREEN(00,00,24,79,MP1)
  RELEASE MP1
  RETURN
```

PDETMOVE.PRG

```
*************************************************************
* PROGRAMA.....: PDETMOVE.PRG
* NOTAS.....: MOVIMIENTO ALTAS Y BAJAS CAMBIOS CONTENIDO DE
*                    PRODUCTOS EN LOS PRESUPUESTOS
* BASE DE DATOS.....: PDETALLE.DBF
* INDEX .....: PDETALLE.NDX (REFE)
* SELECT .....: 3
* FECHA .....: 08-MARZO-1991 -- 29-MARZO-1991
* VERSION .....: 1.0
*************************************************************
MP2 = SAVESCREEN(00,00,24,79)

SELE 3
SET ESCAPE ON
SET EXACT OFF
SET DELE OFF
SET BELL OFF
STORE 0 TO REA,REI,REF,T,REL,IMPORTE,RDA,RDF,RDI
STORE SPACE(8) TO MREFE
STORE SPACE(14) TO BUSC
MNUMERO = "  "
MTIPO = "  "
GO TOP
MRELA = TRIM(MRELA)
IF MRELA = SPACE(8)
  MRELA = "00-00000"
  ?? CHR(7)
ENDIF

FIND &MRELA
DO CASE
  CASE FOUND()
    @ 23,00 CLEAR
    @ 23,00 SAY "ESPERA UN MOMENTO GENERANDO APUNTADORES..."
    REI = RECNO()
    REA = RECNO()
    DO WHI REFE = MRELA
      REF = RECNO()
      SKIP
    ENDD
  CASE .NOT. FOUND() .AND. MRELA="00-00000"
    @ 23,00 CLEAR
    @ 23,00 SAY "ESPERA UN MOMENTO GENERANDO APUNTADORES..."
    APPE BLANK
    REPLACE REFE WITH MRELA
    REPLACE NUMERO WITH "0001"
    REA = RECNO()
    REI = RECNO()
    REF = RECNO()
    KEYBOARD CHR(67)
  CASE .NOT. FOUND() .AND. MRELA = "00-00000"
    GO TOP
    REA = RECNO()
    REI = RECNO()
    GO BOTT
    REF = RECNO()
ENDCASE
***** FORMACION DE PANTALLA DE VISUALIZACION *******
SET COLOR TO N + /W,BG + /N,B + ,,N/BG
CLEAR
SET COLOR TO B + /B
@ 0, 0 SAY REPLICATE("\",80)
SET COLOR TO W + /B
@ 0,33 SAY " PDETALLE.DBF "
SET COLOR TO N + /W
@ 2,13 SAY "REFERENCIA..."
@ 2,26 SAY "[      ]"
@ 1,55 SAY "DESCRIPCION[          ]"
@ 2,55 SAY "TOTAL.....[          ]"
@ 3,1 TO 3,78
@ 4,1 SAY "NUMERO DE"
@ 4,10 TO 10,10
@ 4,11 SAY "-*- D E S C R I P C I O N -*- "
@ 4,46 TO 6,46
@ 4,47 SAY "CANT."
@ 4,52 TO 18,52
@ 4,55 SAY "PRECIO "
@ 4,64 TO 18,64
@ 4,68 SAY "IMPORTE"
@ 5,1 SAY "P L A N O"
@ 6,1 TO 6,78
SET COLOR TO &C_VENTANA
@ 9,4 CLEAR TO 21,76
@ 9,4 TO 21,76
********** LOOP - PRINCIPAL **********
DO WHILE .T.
  IF RECC() # 0
    GO REA
  ENDI
  ********* VISUSLIZACION DEL REGISTRO ACTUAL ******
  SET COLOR TO &C_ENTRA
  @ 2,27 SAY REFE PICTURE "99-99999"
  @ 1,67 SAY RECNO() PICTURE "999,999,999"
  @ 2,67 SAY RECCOUNT() PICTURE "999,999,999"
  @ 7, 2 SAY TIPO
  @ 7, 5 SAY NUMERO
  @ 7,47 SAY CANTIDAD PICTURE "9,999"
  @ 7,53 SAY PRECIO  PICTURE "9999,999.99"
  @ 7,65 SAY IMPORTE PICTURE "999,999,999.99"
  SET COLOR TO N/W
  IF TIPO = "M"
    @ 7,35 SAY " DOLARES "
  ELSE
    @ 7,35 SAY "         "
  ENDIF
  SET COLOR TO &C_VENTANA
  MEMOEDIT(DESC,10,05,20,75,,F,32)
***** AVISO DEL ESTADO ACUAL DEL REGISTRO*********
  SET COLOR TO W/B
  @ 24,00
  @ 24,01 SAY "REG. :"
    @ 24,08 SAY ALLTRIM(STR(RECNO())) + " DE " + ALLTRIM(STR(RECC()))
  DO CASE
    CASE RECC() = 1
      @ 24,23 SAY " 1 SOLO REGISTRO   "
    CASE RECC() = 0
      @ 24,23 SAY " ARCHIVO SIN DATOS  "
    CASE REA = REF
      @ 24,23 SAY " FIN DE ARCHIVO   "
      ?? CHR(7)
    CASE REA = REI
      ?? CHR(7)
```

```
      @ 24,23 SAY " PRINCIPIO DE ARCHIVO"
    ENDC
    IF DELETED()
      SET COLOR TO W + */B
      @ 24,44 SAY " MARCADO "
    ENDIF
    SET COLOR TO N/BG
    **** SELECCION DE LA TECLA DE CONTROL **********
    SET COLOR TO W/B
    @ 23,00
    @ 23,00 SAY "ALTA BUSCAR CAMBIO HOJA TRAER MARCAR
SUPRIMIR "
                                                    *
0123456789012345678901234567890123456789012345678901234567890123
    "          1     2     3     4     5
    SET COLOR TO W + /B
    @ 23,00 SAY "A"
    @ 23,05 SAY "B"
    @ 23,12 SAY "C"
    @ 23,18 SAY "H"
    @ 23,24 SAY "T"
    @ 23,30 SAY "M"
    @ 23,37 SAY "S"
    @ 24,55 SAY CHR(25) + CHR(24) + " PGUP PGDN HOME END
ESC"
    @ 23,78 SAY "
    ************ ACION DE LA TECLA PRESIONADA *********
    T = 0
    DO WHIL T = 0
      T = INKEYTRAP()
    ENDDO
    SET COLOR TO &C_CAJA
    @ 23,00 CLEAR
    DO CASE
      CASE T = 1    &&    TECLA  -  HOME       1 = DB3    26
= DB4
        GO REI
        REA = RECNO()
      CASE T = 6    &&    TECLA   END       6 = DB3    2 = DB4
        GO REF
        REA = RECNO()
      CASE T = 18   &&    TECLA    PGUP
        S = 0
        DO WHILE RECN()REI
          SKIP -1
          REA = RECNO()
          S = S + 1
          IF S = 10
            EXIT
          ENDIF
        ENDDO
      CASE T = 3    &&    TECLA    PGDN
        S = 0
        DO WHILE RECN()REF
          SKIP
          REA = RECNO()
          S = S + 1
          IF S = 10
            EXIT
          ENDIF
        ENDDO
        IF EOF()
          SKIP -1
          REA = RECNO()
        ENDI
      CASE T = 24   &&    TECLA    FLECHA ABAJO
        IF REA = REF
          LOOP
        ELSE
          SKIP 1
          REA = RECNO()
        ENDIF
      CASE T = 5    &&    TECLA    FLECHA ARRIBA
        IF REA = REI
          LOOP
        ELSE
          SKIP -1
          REA = RECNO()
        ENDIF
      CASE T = 27   &&    TECLA    ESC
        EXIT
      CASE CHR(T)$"BB"
        **********BUSQUEDA SEEK FIND LOCATE **********
        BUSC = SPACE(14)
        MREFE = REFE
        MTIPO = TIPO
        MNUMERO = NUMERO
        @ 2,27 GET MREFE PICTURE "99-99999"
        @ 7, 2 GET MTIPO PICTURE "@!" VALID MTIPO$"FEMNIM"
        @ 7, 5 GET MNUMERO PICTURE "9999"
        READ
        IF LASTKEY() = 27
          LOOP
        ELSE
          BUSC = MREFE + MNUMERO + MTIPO
          BUSC = TRIM(BUSC)
          FIND &BUSC
          IF FOUND()
            REA = RECNO()
          ENDI
        ENDI
      CASE CHR(T)$"AACC"
          ************** CAPTURA DE DATOS DEL REGISTO
ACUTUAL ************
        DO WHIL .T.
          SET COLOR TO &C_ENTRA
          IF CHR(T)$"AA"
            GO REF
            MREFE = REFE
            MTIPO = TIPO
            @ 1,67 SAY RECC() + 1 PICTURE "999,999,999"
            @ 2,27 SAY MREFE PICTURE "99-99999"
MTIPO$"FEMNIM"     @ 7, 2 GET MTIPO PICTURE "@!" VALID
            READ
            IF LASTKEY() = 27
              EXIT
            ENDIF
            BUSC = MREFE
            BUSC = TRIM(BUSC)
            FIND &BUSC
            DO WHILE MREFE = REFE
              MNUMERO = NUMERO
              SKIP
            ENDDO
            MNUMERO = RIGHT("0000" + RTRIM(),TRIM(STR(VAL
(MNUMERO) + 1)),4)
            @ 7, 5 GET MNUMERO PICTURE "9999"
            READ
            IF LASTKEY() = 27
              EXIT
            ENDIF
```

```
        BUSC = MREFE + MNUMERO + MTIPO
        FIND &BUSC
        IF FOUND()
          OP = " "
          SET COLOR TO &C_CAJA
          @ 23,00 CLEAR
          @ 23,10 SAY " N# MERO EN PLA NO REPETIDO "
          @ 23,42 SAY MREFE + " " + MTIPO + " " + MNUMERO
          @ 24,10 SAY "DESEAS CONTINUAR /N " 
          @ 24,37 GET OP PICTURE "@!" VALID OP$"SN"
          READ
          IF OP = "N" .OR. LASTKEY() = 27
            LOOP
          ENDIF
        ENDI
        BUSC = MREFE
        BUSC = TRIM(BUSC)
        SELE 1
        FIND &BUSC
        IF .NOT. FOUND()
          OP = " "
          SET COLOR TO &C_CAJA
          @ 23,00 CLEAR
          @ 23,10 SAY "NO HAY REFERENCIA -- "
          @ 23,32 SAY MREFE
          @ 24,10 SAY "DESEAS CONTINUAR /N " 
          @ 24,37 GET OP PICTURE "@!" VALID OP$"SN"
          READ
          IF OP = "N"
            SELE 3
            LOOP
          ENDIF
        ENDIF
        SELE 3
        IF LASTKEY() = 27
          EXIT
        ENDIF
        APPEN BLANK
        REPLACE REFE WITH MREFE
        REPLACE TIPO WITH MTIPO
        REPLACE NUMERO WITH MNUMERO
        REA = RECNO()
        REF = RECNO()
      ENDI
      REA = RECNO()
      SET COLOR TO &C_ENTRA
      @ 1,67 SAY RECNO() PICTURE "999,999,999"
      DO CASE
        CASE CHR(T)$"CC"
          @ 2,27 GET REFE PICTURE "99-99999"
                 @ 7, 2 GET TIPO PICTURE "@!" VALID
TIPO$"EMNSM"
          @ 7, 5 GET NUMERO PICTURE "9999"
        CASE CHR(T)$"AA"
          DO TRAER
          @ 1,67 SAY RECNO() PICTURE "999,999,999"
          @ 2,27 SAY REFE PICTURE "99-99999"
          @ 2,67 SAY RECCOUNT() PICTURE "999,999,999"
          @ 7, 2 SAY TIPO PICTURE "@!"
          @ 7, 5 SAY NUMERO PICTURE "9999"
      ENDCASE
      READ
      SET COLOR TO &C_CAJA
      @ 23,00
      @ 23,00 SAY " CTRL W GRABA Y SALE          ESC
= SALIR SIN GRABAR "
      SET COLOR TO &C_VENTANA
                                    REPLACE DESC WITH
MEMOEDIT(DESC,10,05,20,75,,,70)
      SET COLOR TO &C_CAJA
      @ 23,00
      SET COLOR TO &C_ENTRA
      @ 7,47 GET CANTIDAD PICTURE "9,999"
      @ 7,53 GET PRECIO PICTURE "9999,999.99"
      READ
      MIMPORTE = CANTIDAD*PRECIO
      REPLACE IMPORTE WITH MIMPORTE
      @ 7,65 GET IMPORTE PICTURE "9999,999.99"
      READ
      @ 23,00 CLEAR
      @ 23,00 SAY "ESPERA UN MOMENTO..."
      REA = RECNO()
      MREFE = TRIM(REFE)
      GO TOP
      FIND &MREFE
      IF FOUND()
        REI = RECN()
        DO WHILE REFE = MREFE .AND. .NOT. EOF()
          REF = RECN()
          SKIP
        ENDDO
      ELSE
        SET COLOR TO &C_RESALTA
        @ 23,00
              @ 23,00 SAY "NO SE LOCALIZO REFERENCIA...
CAMBIO DE APUNTADORES."
        INKEYTRAP(0)
        GO TOP
        REI = RECN()
        GO BOTT
        REF = RECN()
      ENDI
      GO REA
      IF CHR(T)$"CC"
        EXIT
      ELSE
        T = ASC("A")
      ENDI
    ENDDO
  CASE CHR(T)$"MM"
    ********* SECCION DELETE Y RECALL **********
    @ 23,00 CLEAR
    @ 23,02 SAY "ESPERA UN MOMENTO..."
    IF DELETED()
      RECALL
    ELSE
      DELETE
    ENDIF
  CASE CHR(T)$"HH"
    ********* VISUALIZACION EN FORMA TABULAR *****
    MPH = SAVESCREEN(01,00,24,79)
    SET COLOR TO &C_CAJA
    MREFE = REFE
    @ 24,00
                                    @ 24,00 SAY " USE
" + CHR(26) + CHR(25) + CHR(27) + CHR(24)
    SET COLOR TO &C_TEXT
    DBEDIT(1,0,23,79)
    RESTSCREEN(01,00,24,79,MPH)
    RELEASE MPH
    IF REFE = MREFE
      REA = RECN()
```

```
            ENDI
            LOOP
         CASE CHR(T)&'TT'
            DO TRAER
         CASE CHR(T)&'SS'
            ***** SECCION PACK Y REINDEX ***************
            SET COLOR TO &C_CAJA
            @ 23,0 CLEAR TO 23,79
            @ 23,10 SAY ' ESPERA UN MOMENTO ELIMINANDO
REGISTROS MARCADOS '
            PACK
            GO TOP
            FIND &MRELA
            IF FOUND()
              REA = RECN()
              REI = RECN()
              DO WHIL REFE = MRELA
                REF = RECNI()
                SKIP
              ENDD
            ELSE
              SET COLOR TO &C_RESALTA
              @ 23,00 CLEAR
              @ 23,00 SAY ' NO EXISTEN CONTENIDOS PARA EL
PRESUPUESTO.  = REGRESA A PRESUPUESTOS.'
              @ 24,00 SAY ' PRESIONA CUALQUIER TECLA PARA
CONTINUAR...'
              INKEYTRAP(X)
              EXIT
            ENDI
         OTHER
            T = 0
            LOOP
      ENDC
   ENDD
   RESTSCREEN(00,00,24,79,MP2)
   RELEASE MP2
   RETURN
```

PIMPPRES.PRG

```
*****************************************************
* PROGRAMA.....: PIMPPRES.PRG
* NOTAS.....: MOVIMIENTO ALTAS Y BAJAS CAMBIOS CONTENIDO DE
*                PRODUCTOS EN LOS PRESUPUESTOS
* BASE DE DATOS.....: PDETALLE.DBF
* INDEX .....: PDETALLE.NDX (REFE)
* SELECT .....: 3
* FECHA .....: 08-MARZO-1991 -- 11-NOV-1991
* VERSION .....: 1.0
*****************************************************
MP3 = SAVESCREEN(00,00,24,79)

STORE 0 TO REA,RELREF,T,PANT,IMP,LLIM,PG,MPIVA
SET ESCAPE ON
SET EXACT OFF
SET DELE ON
SET FUNCTION 10 TO "00-00000" + ";"
STORE 0 TO REA,REI,REF,T,REL,IMPORTE,RDA,RDF,RDI,PANT,IMP,X
STORE 0 TO TOTMN,TOTFE,TOTIM,SUBTOT,IVA,NETO,DESDE,HASTA,LINEA
S,HASTAI,HASTAP
PUBLI MDESCIM,MDESCMN,MDESCFE,MFLETE,MINSTAL,MEXTRAC,MT_CAMBIO
STORE 0 TO MDESCIM,MDESCMN,MDESCFE,MFLETE,MINSTAL,MEXTRAC,MT_CAMBIO,MCOLOC
STORE SPACE(8) TO MREFE
STORE SPACE(14) TO BUSC
IMPRES = "S"
PANT = 22
IMP = 58
MNUMERO = " "
MTIPO = " "
* SELE 5
* TITULO = TITULAR
* SELE 3
GO TOP
FIND &MRELA
SET COLOR TO &C_CAJA
@ 23,00 CLEAR
IF RECC() = 0
  ?? CHR(7)
  ?? CHR(7)
     @ 23,00 SAY 'EL ARCHIVO SE ENCUENTRA VACIO...PRESIONA ALGUNA TECLA.'
  INKEYTRAP(0)
  ?? CHR(18)
  RESTSCREEN(00,00,24,79,MP3)
  RELE MP3
  RETURN
ENDI
MRELA = TRIM(MRELA)
IF MRELA = SPACE(8)
  MRELA = "00-00000"
  ?? CHR(7)
ENDI
FIND &MRELA
SET COLOR TO &C_CAJA
@ 23,00 CLEAR
IF FOUND()
  REA = RECNO()
  REI = RECNO()
  DESDE = 1
  DO WHIL REFE = MRELA
    LINEAS = LINEAS + MLCOUNT(DESC,35) + 3
    REF = RECNO()
    @ 23,02 SAY MRELA
    @ 23,20 SAY REFE
    @ 23,30 SAY RECNO()
    @ 23,40 SAY LINEAS
    SKIP
  ENDD
  HASTAI = INT(((LINEAS + 11)/49) + 2)
  HASTAP = INT(((LINEAS + 11)/13) + 2)
ELSE
  ?? CHR(7)
  ?? CHR(7)
     @ 23,00 SAY 'EL PRESUPUESTO NO TIENE PRODUCTOS...PRESIONA ALGUNA TECLA.'
  INKEYTRAP(0)
  ?? CHR(18)
  RESTSCREEN(00,00,24,79,MP3)
  RELE MP3
  RETURN
ENDI
********************* LOOP - PRINCIPAL *********************
DO WHIL .T.
  STORE 0 TO TOTMN,TOTFE,TOTIM,SUBTOT,IVA,NETO,X,MPIVA
  GO REA
  RESTSCREEN(00,00,24,79,MP3)
  SET COLOR TO &C_CAJA
  @ 23,00 CLEAR
  @ 10,20 CLEAR TO 17,60
  @ 10,20 TO 17,60
  SOMBRA(10,20,17,60)
  SET COLOR TO W + /B
  @ 10,27 SAY ' IMPRESION DEL PRESUPUESTO '
  SET COLOR TO &C_CAJA
     @ 12,35 PROM 'PANTALLA' MESS 'MUESTRA EN PANTALLA EL PRESUPUESTO A IMPRIMIR'
  @ 13,35 PROM 'IMPRIMIR' MESS 'IMPRIMIR EN PAPEL DEL PRESUPUESTO ACTUAL'
  @ 15,35 PROM 'HOJA ' MESS 'EJECUTA UN SALTO DE HOJA EN LA IMPRESORA'
     @ 16,35 PROM 'LIMITES ' MESS 'PERMITE AJUSTAR LOS LIMITES DEL REPORTE'
  MENU TO MOP
  DO CASE
    CASE MOP = 0 .OR. LASTKEY() = 27
       EXIT
    CASE MOP = 1
       SET DEVI TO SCRE
       SET COLOR TO &C_TEXT
       CLEAR
       HASTA = HASTAP
       LIM = PANT
       M1 = 1
       M2 = 10
       M3 = 47
       M4 = 53
       M5 = 55
       M6 = 65
       M7 = 70
       MCM = 35
       MMD = 79
    CASE MOP = 2
```

```
        LIM = IMP
        HASTA = HASTAI
        SET COLOR TO &C_CAJA
        @ 23,00 CLEAR
        DO WHIL ISPRINTER() = .F.
          IMPR = 'R'
          ?? CHR(7) + CHR(7)
          @ 23,00 SAY 'ERROR DE IMPRESORA, VERIFIQUE QUE
ESTE ENCENDIDA'
          @ 24,00 SAY 'DESEAS IMPRIMIR O REGRESAR' GET
IMPR PICT '!' VALID IMPR$'IR'
          READ
          @ 23,00 CLEAR
          IF IMPR = 'R'
            EXIT
          ENDI
        ENDD
        IF ISPRINTER() = .F.
          LOOP
        ENDI
        M1 = 1
        M2 = 15
        M3 = 92
        M4 = 100
        M5 = 105
        M6 = 122
        M7 = 122
        MCM = 75
        MMD = 135
        @ 23,30 SAY 'IMPRIMIENDO...'
        SET DEVI TO PRIN
        @ 0,0 SAY CHR(15)
      CASE MOP = 3
        IF ISPRINTER()
          EJECT
        ENDI
        LOOP
      CASE MOP = 4
        SET COLOR TO &C_CAJA
        @ 23,00 CLEAR
        @ 10,20 CLEAR TO 17,60
        @ 10,20 TO 17,60
        SET COLOR TO W + /B
        @ 10,30 SAY ' LIMITES DE IMPRESION '
        SET COLOR TO &C_CAJA
        @ 12,25 SAY 'DESDE LA PAGINA :'
        @ 13,25 SAY 'HASTA LA PAGINA :'
        @ 14,25 SAY 'HASTA LA PANTALLA :'
        @ 12,45 GET DESDE PICT '999' RANGE 1,HASTAP
        @ 13,45 GET HASTAI PICT '9999' RANGE 1,HASTAI
        @ 14,45 GET HASTAP PICT '9999' RANGE 1,HASTAP
        @ 15,25 SAY 'IMPRIMIR RESUMEN /N:' GET IMPRES PICT
'!' VALID IMPRES$'SN'
        READ
        LOOP
    ENDC
    T = 0
    PG = 0
    LN = 90
    DO WHIL REFE = MRELA
      T = INKEYTRAP()
      IF T = 27
        PAUSA = ' '
        SET PRINT OFF
        SET DEVI TO SCREEN
        @ 23,00 CLEAR
        @ 23,10 SAY 'CONTINUAR /N:' GET PAUSA PICT '!' VALID
PAUSA$'SN'
        READ
        @ 23,00 CLEAR
        IF PAUSA = 'S'
          IF LIM = IMP
            SET PRINT ON
            SET DEVI TO PRINT
          ENDI
        ELSE
          EXIT
        ENDI
      ENDI
      IF LN = LIM
        IF LIM = PANT .AND. (PG1NICIA = DESDE .AND. PG#HASTA)
          SET COLOR TO &C_CAJA
          @ 24,00
          @ 24,00 SAY 'PRESIONA ALGUNA TECLA...'
          INKEYTRAP(0)
          SET COLOR TO &C_TEXT
          CLEAR
          IF LASTKEY() = 27
            EXIT
          ENDI
          CLEAR
        ENDI
        *       IF LIM = PANT
        *         CLEAR
        *       ENDI
        PG = PG + 1
        IF PG = DESDE .AND. PG#HASTA
          @ 0,M7 SAY 'PAG. ' + ALLTRIM(STR(PG))
          IF PG = 1
            SELE 1
            @ 3,M5 SAY 'REF.  ' + REFE
            @ 4,M5 SAY 'FECHA ' + DTOC(FECHA)
            @ 5,M5 SAY 'AGENTE ' + LEFT(AGENTE,15)
            @ 6,M1 SAY REPLICATE('_',MMD)
            @ 7,M1 SAY 'NOMBRE  ' + NOMBRE
            @ 8,M1 SAY 'ATENCION ' + ATENCION
            @ 9,M1 SAY 'DIRECCION ' + LEFT(DIRECCION,60)
            @ 10,M1 SAY 'TELEFONOS ' + TELEFONOS
            @ 11,M1 SAY 'PLAZO DE ENTREGA :' + PLAZO
            @ 12,M1 SAY 'FORMA DE PAGO
:' + LEFT(FORMA1,50)
            @ 13,20 SAY FORMA2
            @ 14,20 SAY FORMA3
            @ 15,M1 SAY 'OBSERVACIONES
:' + LEFT(NOTAS,50)
            @ 16,M1 SAY REPLICATE('_',MMD)
            @ 17,M1 SAY 'NO. EN'
            @ 18,M1 SAY 'PLANO'
            @ 18,M2 SAY 'D E S C R I P C I O N'
            @ 18,M3 SAY 'CANT.'
            @ 18,M4 SAY ' P. UNITARIO'
            @ 18,M6 SAY 'I M P O R T E'
            @ 19,M1 SAY REPLICATE('-',MMD)
            SELE 3
            LN = 20
          ELSE
            SELE 1
            @ 3,M5 SAY 'REF.  ' + REFE
            @ 4,M5 SAY 'FECHA ' + DTOC(FECHA)
            @ 5,M5 SAY 'AGENTE ' + LEFT(AGENTE,15)
            SELE 3
            @ 6,M1 SAY 'NO. EN'
```

```
      @ 7,M1 SAY "PLANO"
      @ 7,M2 SAY "D E S C R I P C I O N"
      @ 7,M3 SAY "CANT."
      @ 7,M4 SAY " P. UNITARIO"
      @ 7,M5 SAY "I M P O R T E"
      @ 8,M1 SAY REPLICATE("-",MMD)
      LN = 9
    ENDI
  ENDI
  IF PG = 1
    LN = 20
  ELSE
    LN = 8
  ENDI
ENDI
IF PG = DESDE .AND. PG<u>HASTA</u>
  @ LN,M1 SAY TIPO + "-" + NUMERO
ENDI

A = MLCOUNT(DESC,MCM)
DO WHIL X<u>A</u>
  X = X + 1
  IF PG = DESDE .AND. PG<u>HASTA</u>
    JUST = MEMOLINE(DESC,MCM,X)
    DO FJUSTIFI WITH JUST,MCM
    @ LN,M2 SAY JUST
  ENDI
  IF X = A .AND. PG = DESDE .AND. PG<u>HASTA</u>
    @ LN,M3 SAY CANTIDAD PICT "9999"
    *       @ LN,50 SAY "NS"
    @ LN,M4 SAY PRECIO PICT "9,999,999.99"
    *       @ LN,65 SAY "NS"
    @ LN,M5 SAY IMPORTE PICT "9,999,999.99"
  ENDI
  IF LN = LIM
    EXIT
  ENDI
  LN = LN + 1
ENDO
IF LN = LIM
  LOOP
ENDI

IF X = A
  DO CASE
    CASE TIPO = "MF"
      TOTMN = TOTMN + IMPORTE
    CASE TIPO = "FE"
      TOTFE = TOTFE + IMPORTE
    CASE TIPO = "IM"
      TOTIM = TOTIM + IMPORTE
  ENDC
  SKIP
  X = 0
  LN = LN + 1
ENDI
ENDO
IF LIM = PANT
  CLEAR
ENDI
SELE 1
PG = PG + 1
IF PG = DESDE .AND. PG<u>HASTA</u>
  @ 0,M7 SAY "PAG. " + ALLTRIM(STR(PG))
  @ 3,M5 SAY "REF.  " + REFE
  @ 4,M5 SAY "FECHA " + DTOC(FECHA)
  @ 5,M5 SAY "AGENTE " + LEFT(AGENTE,15)
  MDESCIM = DESCIM
  MDESCMN = DESCMN
  MDESCFE = DESCFE
  MFLETE = FLETE
  MINSTAL = INSTAL
  MEXTRAC = EXTRAC
  MCOLOCA = COLOCA
  MTCAMBIO = T_CAMBIO
  MPACTO = PACTO
  MDESFINIMP = DESFINIMP
  MDESFINTEX = DESFINTEX
  MPIVA = PIVA
  SELE 3
  LN = 10
  IF TOTIMN<u>O</u>
    @ LN,M2 SAY "SUBTOTAL POR PRODUCTOS DE MARCA NACIONAL"
    @ LN,M5 SAY TOTMN  PICT "99,999,999.99"
    LN = LN + 1
    IF MDESCIMN<u>O</u>
      @ LN,M2 SAY "-"
      @ LN,M2 + 2 SAY MDESCIMN PICT "999.99"
      @ LN,M2 + 9 SAY "% DE DESCUENTO."
      @ LN,M5 SAY (MDESCIMN*TOTIM)/100 PICT "99,999,999.99"
      LN = LN + 1
      @ LN,M2 SAY "TOTAL_____"
      TOTIMN = TOTIMN-(MDESCIMN*TOTIM)/100
      @ LN,M5 SAY TOTIMN  PICT "99,999,999.99"
      LN = LN + 2
    ENDI
    IF LIM = PANT
      SET COLOR TO &C_CAJA
      @ 24,00
      @ 24,00 SAY "PRESIONA ALGUNA TECLA..."
      INKEYTRAP(0)
      SET COLOR TO &C_TEXT
      CLEAR
      LN = 10
    ENDI
  ENDI

  IF TOTFE<u>O</u>
    @ LN,M2 SAY "SUBTOTAL POR PRODUCTOS DE FABRICACI<u>O</u>N ESPECIAL"
    @ LN,M5 SAY TOTFE  PICT "99,999,999.99"
    LN = LN + 1
    IF MDESCFE<u>O</u>
      @ LN,M2 SAY "-"
      @ LN,M2 + 2 SAY MDESCFE PICT "999.99"
      @ LN,M2 + 9 SAY "% DE DESCUENTO."
      @ LN,M5 SAY (MDESCFE*TOTFE)/100 PICT "99,999,999.99"
      LN = LN + 1
      @ LN,M2 SAY "TOTAL_____"
      TOTFE = TOTFE-(MDESCFE*TOTFE)/100
      @ LN,M5 SAY TOTFE  PICT "99,999,999.99"
      LN = LN + 2
    ENDI
    IF LIM = PANT
      SET COLOR TO &C_CAJA
      @ 24,00
      @ 24,00 SAY "PRESIONA ALGUNA TECLA..."
      INKEYTRAP(0)
```

```
        SET COLOR TO &C_TEXT
        CLEAR
        LIN = 10
      ENDI
    ENDI

    IF MFLETEO
      @ LIN,M2 SAY "POR CONCEPTO DE FLETE "
      @ LIN,M6 SAY MFLETE PICT "99,999,999.99"
      LIN = LIN + 1
    ENDI
    IF MEXTRACO
      @ LIN,M2 SAY "POR CONCEPTO DE EXTRACCI'N"
      @ LIN,M6 SAY MEXTRAC PICT "99,999,999.99"
      LIN = LIN + 1
    ENDI
    IF MINSTALO
      @ LIN,M2 SAY "POR CONCEPTO DE INSTALACION"
      @ LIN,M6 SAY MINSTAL PICT "99,999,999.99"
      LIN = LIN + 1
    ENDI
    IF MCOLOCAO
      @ LIN,M2 SAY "POR CONCEPTO DE COLOCACI'N"
      *      @ LIN,64 SAY "NS"
      @ LIN,M6 SAY MCOLOCA PICT "99,999,999.99"
      LIN = LIN + 1
    ENDI

    LIN = LIN + 2
    @ LIN,M1 SAY "SUBTOTAL POR PRODUCTOS NACIONALES
Y DE FABRICACION ESPECIAL"
    @ LIN,M1 SAY "SUBTOTAL POR PRODUCTOS NACIONALES
Y DE FABRICACION ESPECIAL"
    SUBTOT = 0
    SUBTOT = MCOLOCA + MINSTAL + MEXTRAC + MFLETE +
TOTFE + TOTMN
    *      @ LIN,64 SAY "NS"
    @ LIN,M6 SAY SUBTOT PICT "99,999,999.99"
    LIN = LIN + 1
    IF MDESFINIMPO
      @ LIN,M2 SAY "-"
      @ LIN,M2 + 2 SAY MDESFINIMP PICT "999.99"
      @ LIN,M2 + 9 SAY "%" + MDESFINTEX
            @ LIN,M6 SAY (SUBTOT*MDESFINIMP)/100 PICT
"99,999,999.99"
      SUBTOT = SUBTOT-(SUBTOT*MDESFINIMP)/100
      LIN = LIN + 1
      @ LIN,M2 SAY "TOTAL....."
      *      @ LIN,64 SAY "NS"
      @ LIN,M6 SAY SUBTOT PICT "99,999,999.99"
      LIN = LIN + 1
    ENDI
    IF MPACTOO
      @ LIN,M2 SAY "-"
      @ LIN,M2 + 2 SAY MPACTO PICT "999.99"
      @ LIN,M2 + 9 SAY "% DE DESCUENTO POR PACTO"
            @ LIN,M6 SAY (SUBTOT*MPACTO)/100 PICT
"99,999,999.99"
      SUBTOT = SUBTOT-(SUBTOT*MPACTO)/100
      LIN = LIN + 1
      @ LIN,M2 SAY "TOTAL....."
      *      @ LIN,64 SAY "NS"
      @ LIN,M6 SAY SUBTOT PICT "99,999,999.99"
      LIN = LIN + 1
    ENDI
    IVA = SUBTOT*MPIVA/100
    NETO = SUBTOT + IVA
    @ LIN,M2 SAY ALLTRIM(STR(MPIVA)) + "% DE I.V.A."
    @ LIN,M6 SAY IVA PICT "99,999,999.99"
    LIN = LIN + 1
    @ LIN,M2 SAY "TOTAL ....."
    @ LIN,M6 SAY NETO PICT "99,999,999.99"
    LIN = LIN + 1

    IF LIM = PANT
      SET COLOR TO &C_CAJA
      @ 24,00
      @ 24,00 SAY "PRESIONA ALGUNA TECLA..."
      INKEYTRAP(0)
      SET COLOR TO &C_TEXT
      CLEAR
      LIN = 10
    ENDI

    IF TOTMO
      LIN = LIN + 2
        @ LIN,M1 SAY "SUBTOTAL POR PRODUCTOS DE
IMPORTACION EN (DOLARES)"
        @ LIN,M1 SAY "SUBTOTAL POR PRODUCTOS DE
IMPORTACION EN (DOLARES)"
        *      @ LIN,64 SAY "NS"
      @ LIN,M6 SAY TOTIM PICT "99,999,999.99"
      LIN = LIN + 1
      IF MDESCIMO
        @ LIN,10 SAY "-"
        @ LIN,M2 + 2 SAY MDESCIM PICT "999.99"
        @ LIN,M2 + 9 SAY "% DE DESCUENTO."
              @ LIN,M6 SAY (MDESCIM*TOTIM)/100 PICT
"99,999,999.99"
        LIN = LIN + 1
        @ LIN,M2 SAY "TOTAL ....."
        TOTIM = TOTIM-(MDESCIM*TOTIM)/100
        @ LIN,M6 SAY TOTIM PICT "99,999,999.99"
        LIN = LIN + 1
        IF MTCAMBIOO
          @ LIN,M2 SAY "AL TIPO DE CAMBIO :"
          @ LIN,M2 + 30 SAY MTCAMBIO PICT "@$ 999.99"
          TOTIM = TOTIM*MTCAMBIO
          @ LIN,M6 SAY TOTIM PICT "99,999,999.99"
          LIN = LIN + 1
          IF MDESFINIMPO
            @ LIN,M2 SAY "-"
            @ LIN,M2 + 2 SAY MDESFINIMP PICT "999.99"
            @ LIN,M2 + 9 SAY "% " + MDESFINTEX
            *              @ LIN,64 SAY "NS"
                  @ LIN,M6 SAY (TOTIM*MDESFINIMP)/100 PICT
"99,999,999.99"
            TOTIM = TOTIM-(TOTIM*MDESFINIMP)/100
            LIN = LIN + 1
            @ LIN,M2 SAY "TOTAL....."
            @ LIN,M6 SAY TOTIM PICT "99,999,999.99"
            LIN = LIN + 1
          ENDI
          IF MPACTOO
            @ LIN,M2 SAY "-"
            @ LIN,M2 + 2 SAY MPACTO PICT "999.99"
            @ LIN,M2 + 9 SAY "% DE DESCUENTO POR PACTO"
                  @ LIN,M6 SAY (TOTIM*MPACTO)/100 PICT
"99,999,999.99"
            TOTIM = TOTIM-(TOTIM*MPACTO)/100
            LIN = LIN + 1
            @ LIN,M2 SAY "TOTAL....."
            @ LIN,M6 SAY TOTIM PICT "99,999,999.99"
            LIN = LIN + 1
```

```
            ENDI
            @ LN,M2 SAY ALLTRIM(STR(MPIVA)) + "% DE I.V.A."
            @ LN,M6 SAY TOTIM*MPIVA/100 PICT "99,999,999.99"
            LN = LN + 1
            @ LN,M2 SAY "TOTAL....."
            TOTIM = TOTIM + (TOTIM*MPIVA/100)
            @ LN,M6 SAY TOTIM PICT "99,999,999.99"
          ELSE
            @ LN,M2 SAY ALLTRIM(STR(MPIVA)) + "% DE I.V.A."
            @ LN,M6 SAY TOTIM*MPIVA/100 PICT "99,999,999.99"
            LN = LN + 1
            @ LN,M2 SAY "TOTAL (DOLARES)....."
            TOTIM = TOTIM + (TOTIM*MPIVA/100)
            @ LN,M6 SAY TOTIM PICT "99,999,999.99"
          ENDI
        ELSE
          IF MTCAMBIO
            @ LN,M2 SAY "AL TIPO DE CAMBIO :"
            @ LN,M2 + 30 SAY MTCAMBIO PICT "@$ 999.99"
            TOTIM = TOTIM*MTCAMBIO
            @ LN,M6 SAY TOTIM PICT "99,999,999.99"
            LN = LN + 1
            IF MDESFINIMP0
              @ LN,M2 SAY "-"
              @ LN,M2 + 2 SAY MDESFINIMP PICT "999.99"
              @ LN,M2 + 9 SAY "% " + MDESFINTEX
                    @ LN,M6 SAY (TOTIM*MDESFINIMP)/100 PICT
"99,999,999.99"
              TOTIM = TOTIM- (TOTIM*MDESFINIMP)/100
              LN = LN + 1
              @ LN,M2 SAY "TOTAL....."
              @ LN,M6 SAY TOTIM PICT "99,999,999.99"
              LN = LN + 1
            ENDI
            IF MPACTO0
              @ LN,M2 SAY "-"
              @ LN,M2 + 2 SAY MPACTO PICT "999.99"
              @ LN,M2 + 8 SAY "% DE DESCUENTO POR PACTO"
                    @ LN,M6 SAY (TOTIM*MPACTO)/100 PICT
"99,999,999.99"
              TOTIM = TOTIM- (TOTIM*MPACTO)/100
              LN = LN + 1
              @ LN,M2 SAY "TOTAL....."
              @ LN,M6 SAY TOTIM PICT "99,999,999.99"
              LN = LN + 1
            ENDI
            @ LN,M2 SAY ALLTRIM(STR(MPIVA)) + "% DE I.V.A."
            @ LN,M6 SAY TOTIM*MPIVA/100 PICT "99,999,999.99"
            LN = LN + 1
            @ LN,M2 SAY "TOTAL....."
            TOTIM = TOTIM + (TOTIM*MPIVA/100)
            @ LN,M6 SAY TOTIM PICT "99,999,999.99"
          ELSE
            @ LN,M2 SAY ALLTRIM(STR(MPIVA)) + "% DE I.V.A."
            @ LN,M6 SAY TOTIM*MPIVA PICT "99,999,999.99"
            LN = LN + 1
            @ LN,M2 SAY "TOTAL (DOLARES)....."
            TOTIM = TOTIM + (TOTIM*MPIVA/100)
            @ LN,M6 SAY TOTIM PICT "99,999,999.99"
          ENDI
        ENDI
      ENDI

      IF LIM = PANT
        LN = 16
        SET COLOR TO &C_CAJA
        @ 24,00
        @ 24,00 SAY "PRESIONA ALGUNA TECLA..."
        INKEYTRAP(0)
        SET COLOR TO &C_TEXT
        CLEAR
      ENDI
      LN = LN + 4
      IF MTCAMBIO
        FINAL = TOTIM + NETO
        @ LN,M1 SAY "IMPORTE TOTAL POR PRODUCTOS DEL
PRESUPUESTO : " + MRELA
        @ LN,M1 SAY "IMPORTE TOTAL POR PRODUCTOS DEL
PRESUPUESTO : " + MRELA
        @ LN,M1 SAY "IMPORTE TOTAL POR PRODUCTOS DEL
PRESUPUESTO : " + MRELA
        @ LN,M6-2 SAY "N$"
        @ LN,M6 SAY FINAL PICT "99,999,999.99"
        @ LN,M6 SAY FINAL PICT "99,999,999.99"
      ENDI
    ENDI
    @ LN,0 SAY CHR(18)
    ?? CHR(18)
    IF IMPRESS"SS"
      SELE 5
      LN = 10
      X = 0
      A = MLCOUNT(RESUMEN,65)
      DO WHIL X<A
        X = X + 1
        JUST = MEMOLINE(RESUMEN,65,X)
        DO PJUSTIFI WITH JUST,65
        @ LN,M2 SAY JUST
        IF LN = LIM
          LN = 10
          IF LIM = PANT
            @ 24,00
            @ 24,00 SAY "PRESIONA ALGUNA TECLA PARA
CONTINUAR..."
            INKEYTRAP(0)
            CLEAR
          ENDI
        ENDI
        LN = LN + 1
      ENDD
      SELE 3
      IF LIM = IMP
        EJECT
      ELSE
        @ 24,00
        @ 24,00 SAY "PRESIONA ALGUNA TECLA PARA
CONTINUAR..."
        INKEYTRAP(0)
        CLEAR
      ENDI
    ENDI

    SET DEVI TO SCRE
    SET PRINT OFF
    SELE 3
                                        STORE    0    TO
TOTMN,TOTFE,TOTIM,SUBTOT,IVA,NETO,X,MPIVA
    GO REI

    IF LIM = PANT
      SET COLOR TO &C_CAJA
      @ 24,00
      @ 24,00 SAY "FIN DEL PRESUPUESTO. PRESIONA ALGUNA
TECLA..."
```

```
    INKEYTRAP(0)
    SET COLOR TO &C_TEXT
    CLEAR
  ENDI
  LOOP
  * - - - - - - - - - - - - - - - - - - - - - - - -
  * - - - - - FINAL DEL REPORTE - - - - - - - - - -
ENDD
?? CHR(18)
RESTSCREEN(00,00,24,79,MP3)
RELEASE MP3
RETU
*
```

PBASMOVE.PRG

```
*****************************************************
* PROGRAMA.....: PBASMOVE.PRG
* NOTAS.....: MOVIMIENTO ALTAS Y BAJAS CAMBIOS A CATALOGO
* PROGRAMA.....: PBASMOVE.PRG
* BASE DE DATOS.....: PBASICOS.DBF
* INDEX .....: PBASICOS.NDX
* SELECT .....: 2
* VERSION .....: 2.0 (CLIPPER)
* REALIZADO POR : ARTURO CROTTE FRANCO
*****************************************************
MP3 = SAVESCREEN(00,00,24,79)
SAVESCREEN(00,00,24,79,MP3)
 SELE 2
SET SAFE OFF
SET ESCAPE ON
SET EXACT OFF
SET DELE OFF
SET BELL OFF
STORE 0 TO REA,REI,REF,T,REL,MIMPORTE
STORE SPACE(6) TO MREFE
STORE SPACE(14) TO BUSC
MNUMERO = "  "
MTIPO = " "
GO TOP
REI = RECN()
REA = RECN()
GO BOTT
REF = RECN()
******* FORMACION DE PANTALLA DE VISUALIZACION *******
SET COLOR TO B+/B
@ 0, 0 SAY REPLICATE("\",80)
SET COLOR TO W+/B
@ 0,20 SAY " CATLOGO DE PRODUCTOS BSICOS O DE LINEA "
SET COLOR TO &C_TEXT
@ 1,00 CLEAR
@ 2,11 SAY "REGISTRO ACTUAL....."
@ 3,11 SAY "TOTAL DE REGISTROS..."
@ 4,02 TO 4,77
@ 5,19 TO 18,19
@ 5,50 TO 18,50
@ 7,02 TO 7,50
@ 4,19 SAY "B"
@ 4,50 SAY "I"
@ 7,19 SAY "E"
@ 7,50 SAY "4"
@ 5,03 SAY "NO. EN"
@ 5,25 SAY " D E S C R I P C I O N "
@ 3,51 SAY "   PRECIO - UNITARIO   "
@ 4,51 SAY "  COMPAAIA -COMPETIDORES"
@ 6,03 SAY "LISTA "
@ 6,25 SAY "D E L  P R O D U C T O"
SET COLOR TO &C_VENTANA
@ 9,4 CLEAR TO 21,76
@ 9,4 TO 21,76
*************** LOOP - PRINCIPAL **********************
DO WHILE .T.
  IF RECC() # 0
    GO REA
  ENDI
********* VISUSLIZACION DEL REGISTRO ACTUAL *******
  SET COLOR TO &C_ENTRA
  @ 02,32 SAY RECN() PICTURE "9,999,999"
  @ 03,32 SAY RECC() PICTURE "9,999,999"
  @ 08,02 SAY TIPO
  @ 08,05 SAY LISTA PICTURE "9999"
  @ 08,51 SAY PRECIO PICTURE "9999,999.99"
  @ 05,55 SAY COMPE1 PICTURE "9999,999.99"
  @ 06,55 SAY COMPE2 PICTURE "9999,999.99"
  @ 07,55 SAY COMPE3 PICTURE "9999,999.99"
  @ 08,55 SAY COMPE4 PICTURE "9999,999.99"
  SET COLOR TO N/W
  IF TIPO = "IM"
    @ 8,40 SAY " DOLARES "
  ELSE
    @ 8,40 SAY "         "
  ENDIF
  SET COLOR TO &C_VENTANA
  MEMOEDIT(DESC,10,05,20,75,,F,,32)
  IF OPERA = "1"
    SET COLOR TO &C_RESALTA
    @ 8,21 SAY " OPERA  =  1 "
  ELSE
    SET COLOR TO &C_TEXT
    @ 8,21 SAY "         "
  ENDI
  ******AVISO DEL ESTADO ACUAL DEL REGISTRO**********
  SET COLOR TO &C_CAJA
  @ 24,00
  @ 24,01 SAY "REG. :"
          @ 24,08 SAY ALLTRIM(STR(RECN())) + " DE " + ALLTRIM(STR(RECC()))
  DO CASE
    CASE RECC() = 1
       @ 24,23 SAY " 1 SOLO REGISTRO    "
    CASE RECC() = 0
       @ 24,23 SAY " ARCHIVO SIN DATOS  "
    CASE REA = REF
       @ 24,23 SAY " FIN DE ARCHIVO     "
    CASE REA = REI
       @ 24,23 SAY " PRINCIPIO DE ARCHIVO"
  ENDC
  IF DELETED()
    SET COLOR TO W+/B
    @ 24,44 SAY " MARCADO "
  ENDIF
  SET COLOR TO N/BG
  ************** SELECCION DE LA TECLA DE CONTROL ****
  SET COLOR TO W/B
  @ 23,00
  @ 23,00 SAY "ALTA BUSCAR CAMBIO HOJA TRAER MARCAR OPERA "
    SET COLOR TO W+/B
  @ 23,00 SAY "A"
  @ 23,05 SAY "B"
  @ 23,12 SAY "C"
  @ 23,18 SAY "H"
  @ 23,24 SAY "T"
  @ 23,30 SAY "M"
  @ 23,38 SAY CHR(17) + "DY"
  @ 24,55 SAY CHR(25) + CHR(24) + " PGUP PGDN HOME END ESC"
  @ 23,78 SAY "
  ********** ACION DE LA TECLA PRESIONADA ************
  T = 0
```

```
@ 23,76 SAY ""
DO WHIL T = 0
 T = INKEYTRAP()
ENDDO
SET COLOR TO &C_CAJA
@ 23,00 CLEAR
DO CASE
  CASE T = 1    &&   TECLA   HOME      1 = DB3   26
= DB4
    GO REI
    REA = RECNO()
  CASE T = 6    &&   TECLA   END       6 = DB3   2 = DB4
    GO REF
    REA = RECNO()
  CASE T = 18   &&   TECLA   PGUP
    S = 0
    DO WHILE RECNO()<REI
     SKIP -1
     REA = RECNO()
     S = S + 1
     IF S = 10
       EXIT
     ENDIF
    ENDDO
  CASE T = 3    &&   TECLA   PGDN
    S = 0
    DO WHILE RECNO()<REF
     SKIP
     REA = RECNO()
     S = S + 1
     IF S = 10
       EXIT
     ENDIF
    ENDDO
    IF EOF()
     SKIP -1
     REA = RECNO()
    ENDI
  CASE T = 24   &&   TECLA   FLECHA ABAJO
    IF REA = REF
     LOOP
    ELSE
     SKIP 1
     REA = RECNO()
    ENDIF
  CASE T = 5    &&   TECLA   FLECHA ARRIBA
    IF REA = REI
     LOOP
    ELSE
     SKIP -1
     REA = RECNO()
    ENDIF
  CASE T = 27   &&   TECLA   ESC
    EXIT
  CASE CHR(T)$"BB"
    ********** BUSQUEDA SEEK,FIND LOCATE *********
    MTIPO = TIPO
    MUSTA = LISTA
   @ 8,02 GET MTIPO PICTURE "@!" VALID MTIPO$"FEMNIM"
    @ 8,05 GET MUSTA PICTURE "9999"
    READ
    IF LASTKEY() = 27
     LOOP
    ELSE
     BUSC = "    "
     BUSC = MTIPO + MUSTA
     FIND &BUSC
     IF FOUND()
      REA = RECNO()
     ELSE
      LOOP
     ENDIF
    ENDIF
  CASE CHR(T)$"AACC"
    ******* CAPTURA DE DATOS DEL REGISTO ACUTUAL **
    SET COLOR TO &C_ENTRA
    IF CHR(T)$"AA"
     GO REF
     MTIPO = TIPO
     MUSTA = RIGHT("0000" + RTRIM(LTRIM(STR(VAL(LIST
A) + 1))),4)
     @ 2,32 SAY RECCOUNT() + 1 PICTURE "9,999,999"
    ELSE
     @ 8,02 GET MTIPO PICTURE "@!" VALID
MTIPO$"FEMNIM"
     @ 8,05 GET MUSTA PICTURE "9999"
     READ
     IF MUSTA = "0000" .OR. LASTKEY() = 27
      EXIT
     ENDIF
     BUSC = "    "
     BUSC = MTIPO + MUSTA
     FIND &BUSC
     IF FOUND()
      MOP = " "
      SET COLOR TO &C_CAJA
      @ 23,00 CLEAR
      @ 23,00 SAY " ESTE REGISTRO YA EXISTE ~
" + MTIPO + "-" + MUSTA
      @ 24,10 SAY "DESEAS CONTINUAR  /N "
      @ 24,37 GET OP PICTURE "!" VALID MOP$"SN"
      READ
      @ 23,00 CLEAR
      IF MOP = "N"
       LOOP
      ENDIF
      IF LASTKEY() = 27
       LOOP
      ENDIF
     ENDI
     APPE BLAN
     REPLACE TIPO WITH MTIPO
     REPLACE LISTA WITH MUSTA
    ENDI
    REA = RECNO()
    SET COLOR TO &C_ENTRA
    @  1,67 SAY RECNO() PICTURE "999,999,999"
    IF CHR(T)$"CC"
     @ 8,02 GET TIPO PICTURE "@!" VALID TIPO$"FEMNIM"
     @ 8,05 GET LISTA PICTURE "9999"
     @ 2,32 SAY RECNO() PICTURE "9,999,999"
     @ 3,32 SAY RECCOUNT() PICTURE "9,999,999"
    ELSE
     @ 8,02 SAY TIPO PICTURE "@!"
     @ 8,05 SAY LISTA PICTURE "9999"
     @ 2,32 SAY RECNO() PICTURE "9,999,999"
     @ 3,32 SAY RECCOUNT() PICTURE "9,999,999"
    ENDI
    READ
    SET COLOR TO &C_CAJA
    @ 23,00
```

```
                @ 23,00 SAY '  CTRL W= GRABA Y SALE
ESC = SALIR SIN GRABAR '
        SET COLOR TO &C_VENTANA
                                    REPLACE  DESC  WITH
MEMOEDIT(DESC,10,05,20,75,,T,,"70)
        SET COLOR TO &C_CAJA
        @ 23,00
        SET COLOR TO &C_ENTRA
        @ 06,51 GET PRECIO PICTURE "9999,999.99"
        @ 05,65 GET COMPE1 PICTURE "9999,999.99"
        @ 06,65 GET COMPE2 PICTURE "9999,999.99"
        @ 07,65 GET COMPE3 PICTURE "9999,999.99"
        @ 08,65 GET COMPE4 PICTURE "9999,999.99"
        READ
        @ 23,00 CLEAR
        @ 23,00 SAY "ESPERA UN MOMENTO..."
        REA = RECNO()
        GO TOP
        REI = RECNO()
        GO BOTT
        REF = RECNO()
      CASE CHR(T)$"Mm"
        ******* SECCION DELETE Y RECALL **************
        @ 23,00 CLEAR
        @ 23,02 SAY "ESPERA UN MOMENTO..."
        IF DELETED()
          RECALL
         ELSE
          DELETE
        ENDIF
      CASE CHR(T)$"Hh"
        ******** VISUALIZACION EN FORMA TABULAR *******
        MPH = SAVESCREEN(01,00,24,79)
        SAVESCREEN(01,00,24,79,MPH)
        SET COLOR TO &C_CAJA
        @ 24,00
                                    @  24,00  SAY  '  USE
' + CHR(26) + CHR(25) + CHR(27) + CHR(24)
        SET COLOR TO &C_TEXT
        DBEDIT(1,0,23,79)
        RESTSCREEN(01,00,24,79,MPH)
        REA = RECNO()
        LOOP
      CASE CHR(T)$"Tt"
        DO TRAERBAS
      CASE T = 13
            ********** TECLA ENTER PARA SELECCION
ESPECIAL **********
        IF OPERA = "1"
          REPLACE OPERA WITH "0"
         ELSE
          REPLACE OPERA WITH "1"
        ENDI
      OTHER
        T = 0
        LOOP
    ENDC
  ENDD
  IF MREL = 1
    MPASOMEMO = DESC
    MPRECIO = PRECIO
  ENDI
  RESTSCREEN(00,00,24,79,MP3)
  RETURN
```

PACCESOS.PRG

```
*************************************************
*
* VISUALIZACION DE ACCESOS AL SISTEMA
*
* SELE 1
* BASE : ACCESOS
*
*************************************************
MPM2 = SAVESCREEN(01,00,24,79)

SELE 1
USE PACCESOS
GO BOTT
SET COLOR TO &C_CAJA
@ 00,00
@ 00,00 SAY ' LISTA DE USUARIOS QUE HAN ACCESADOS AL
SISTEMA'
@ 24,00
                    @          24,00       SAY       '          USE
' + CHR(26) + CHR(25) + CHR(27) + CHR(24) + ' HOME CONSUF
O
 ' SALIR '
SET COLOR TO &C_TEXT
@ 1,0 TO 23,78 DOUB
DBEDIT(2,1,22,78)
RESTSCREEN(01,00,24,79,MPM2)
RELE MPM2
USE
RETU
```

BIBLIOGRAFIA

1.- DONALD D. SANDERS

INFORMATICA PRESENTE Y FUTURO

Mc GRAW HILL

TERCERA EDICION

pp 43-50, 855-867

2.- EDWARD JONES

APLIQUE DBASE IV

Mc GRAW HILL

pp 1-13, 397-423

3.- FRANCISCO MARIN QUIROZ

CLIPPER SUMMER 87

MACROBIT

pp 7-48

4.- FRANK THRUN

CLIPPER 5.0 ACCESO RAPIDO

COMPUTEC EDITORES

pp 65-88

5.- JAMES A. SENN

ANALISIS Y DISEÑO DE SISTEMAS DE INFORMACION

Mc GRAW HILL

pp 4-34

6.- JOSE A. RAMALHO

CLIPPER 5.0 AVANZADO

Mc GRAW HILL

pp 1-15

7.- JOSE JAVIER GARCIA-BADELL

CLIPPER 5.0

GUIA DEL COMPILADOR PARA DBASE III Y IV

Mc GRAW HILL

pp 7-34

8.- MICROSOFT WINDOWS 3.1

GUIA DE USUARIO

MICROSOFT

pp 485-505