I+91
11904
T-173.
Pt. 2

ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

MARCADOR TELEFONICO AUTOMATICO

ANEXO 2

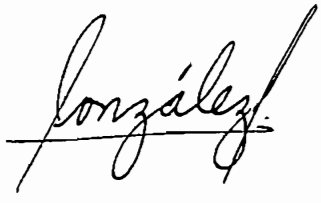
LISTADOS DE PROGRAMAS

TESIS PREVIA A LA OBTENCION DEL TITULO DE INGENIERO EN LA ESPECIALIDAD DE ELECTRONICA Y TELECOMUNICACIONES

HENDERSON ALEXANDER ITURRALDE PASQUEL

QUITO, JUNIO DE 1996

Certifico que los siguientes programas han sido realizados en su totalidad por el señor Henderson Iturralde.



Ing. Fabio González.

INDICE

A2.1 Listado del programa para el microcontrolador.

A2.2 Listado del programa para el computador personal.

anexo 2.1

Listado del programa para el microcontrolador

```
;       ================================================================
;               p6042908.txt
;               NOSTROMO
;               MARCADOR TELEFONICO AUTOMATICO
;               directorio de 350 abonados
;       atencion a la linea en tonos y pulsos
;       llamada entrante (el uC como monoestable redisparable con intx)
;       equipo normalmente encendido y normalmente apagado
;       equipo activo
;       modo directorio: busqueda y marcado; directorio, hot, abreviado
;       modo programacion: agregar, eliminar, hot line, marc abr,
;                          linea-equipo, tiempo
;       reloj, cronometro: (llout only mientras la otra linea no ocupe el
;                           equipo)
;       transmision - recepcion
;       ================================================================


;       ********        banderas
f_l1            EQU     00H     ; add/datos
f_l2            EQU     01H     ; lcd nibble
f_l3            EQU     02H     ; escrita linea 1
f_l4            EQU     03H     ; escrito msnibble del numero de ram
f_rm            EQU     04H     ; s_lcd_rom con retardo maximo
f_r1            EQU     05H     ; s_lcd_rom con retardo = 1 seg
f_lcdabrp       EQU     06H     ; mensaje abreviados en modo programacion
f_eaua          EQU     07H     ; existe almenos un abreviado

f_att1          EQU     08H     ; atendida llamada en curso por lt1
f_att2          EQU     09H     ; atendida llamada en curso por lt2
f_lt2att        EQU     0AH     ; lt2 esta siendo atendida
f_alf           EQU     0DH     ; tecla de navegacion o alfabeto
f_k1            EQU     0EH     ; fin lectura 3 primeras teclas
f_k2            EQU     0FH     ; tecla anterior = 1

f_cro1          EQU     11H     ; cronometrar lt1
f_cro2          EQU     12H     ; cronometrar lt2
f_hex           EQU     13H     ; cargar hora actual en hexadecimal
f_nf            EQU     14H     ; grupo sin abonados
f_cndc          EQU     15H     ; definir al abonado como normal (en s_fndc)
f_fndc          EQU     16H     ; nibble de configuracion encontrado
f_clraut        EQU     17H     ; =1 realizando limpieza automatica

f_ldi           EQU     18H     ; ldi ejecutada
f_ldn           EQU     19H     ; ldn ejecutada
f_on            EQU     1AH     ; codigo de acceso ejecutado
f_cel           EQU     1BH     ; llamada a celular
f_tpni          EQU     1DH     ; numero en el bloque con bloqueo
f_elim          EQU     1EH     ; s_ehm para eliminar
f_hotl          EQU     1FH     ; s_ehm para definir hot line

f_dp            EQU     20H     ; parte alta / baja del directorio
f_dpbk          EQU     21H     ; parte alta / baja del directorio (respaldo)
f_mar           EQU     22H     ; marcar numero telefonico
f_na            EQU     24H     ; volver a modo normalmente apagado

f_ti            EQU     28H     ; trabajando con teclado interno
f_tecla1        EQU     29H     ; teclado interno habil: 1
f_tecla3        EQU     2AH     ; teclado interno habil: 3
f_tpos          EQU     2CH     ; posibilidades de la tecla (=0:3; =1:4)

                                ; flagic0 = flagxc0 : configuracion en rami
f_pt1           EQU     30H     ; lt1 pulsos / tonos
f_bi1           EQU     31H     ;     bloqueo ldi
f_bn1           EQU     32H     ;     bloqueo ldn
f_pt2           EQU     33H     ; lt2 pulsos / tonos
f_bi2           EQU     34H     ;     bloqueo ldi
f_bn2           EQU     35H     ;     bloqueo ldn
f_bt2           EQU     36H     ;     bloqueo total
f_ne            EQU     37H     ; equipo normalmente encendido

                                ; banco de reles
r_lt1           EQU     48H     ; rele lt1 bloqueo
r_dr1           EQU     49H     ;          tonos
```

```
r_flt1          EQU     4AH     ;           filtro
r_llin1         EQU     4BH     ;           llin 1
r_lt2           EQU     4CH     ; rele lt2 bloqueo
r_dr2           EQU     4DH     ;           tonos
r_flt2          EQU     4EH     ;           filtro
r_llin2         EQU     4FH     ;           llin 2

;       ********        puerto 1
ix_lt1          EQU     90H     ; lt1 ocupada
ix_tel1         EQU     91H     ; telf1 descolgado
ix_lt2          EQU     92H     ; lt2 ocupada
ix_tel2         EQU     93H     ; telf2 descolgado
ix_beep         EQU     94H     ; control de beeper
ix_lv           EQU     95H     ; llave
ix_sw1          EQU     96H     ; pulsante 1
ix_ser          EQU     97H     ; cable serial conectado

;       ********        puerto 3
ix_llin1        EQU     0B2H    ; senal de timbre en lt1
ix_llin2        EQU     0B3H    ; senal de timbre en lt2
ix_dtmf         EQU     0B4H    ; DV SSI75T202
ix_425          EQU     0B5H    ; 425Hz detectado

;       ********        acumulador
A               EQU     0E0H    ; acumulador (para push y pop)
d_dim           EQU     0E4H    ; decena del dia del mes

dis_rs          EQU     0E4H    ; bit de control del display: tipo de dato
dis_e           EQU     0E6H    ; bit de control del display: habilitacion

;       ********        memoria
cllin1          EQU     08H     ; contador para llamada entrante 1
cllin2          EQU     09H     ; contador para llamada entrante 2
ss1             EQU     0AH     ; segundo inicio llout lt1
mm1             EQU     0BH     ; minuto inicio llout lt1
hh1             EQU     0CH     ; hora inicio llout lt1
ss2             EQU     0DH     ; segundo inicio llout lt2
mm2             EQU     0EH     ; minuto inicio llout lt2
hh2             EQU     0FH     ; hora inicio llout lt2

dir_pun         EQU     12H     ; puntero del directorio
dir_punbk       EQU     13H     ; puntero del directorio (respaldo)
stk_dpl         EQU     14H     ; stack DPL
stk_dph         EQU     15H     ; stack DPH
tec_pre         EQU     16H     ; tecla presionada
grupo           EQU     17H     ; ASCII del grupo

centena         EQU     18H     ; centena del numero del abonado
dec_uni         EQU     19H     ; decena - unidad del numero del abonado
val_ndc         EQU     1BH     ; valor del nibble de configuracion
lcd_pun         EQU     1EH     ; puntero del display
lcd_dat         EQU     1FH     ; dato a ser escrito en el display

flagic0         EQU     26H     ; banderas de config en rami
ban_rele        EQU     29H     ; banco de control de reles
nombre          EQU     30H     ; puntero del nombre
numero          EQU     40H     ; puntero del numero

ssa             EQU     40H     ; segundo actual
mma             EQU     41H     ; minuto actual
hha             EQU     42H     ; hora actual
ssd             EQU     43H     ; segundo para sacar a display
mmd             EQU     44H     ; minuto para sacar a display
hhd             EQU     45H     ; hora para sacar a display

;       ********        ram externa
ramx1           EQU     2000H   ; display
ramx3           EQU     6000H   ; dtmf rd (ssi75t202)
ramx4           EQU     8000H   ; dtmf wr (tms5087m)
```

```
ramx5           EQU     0A000H   ; reles

c_seg           EQU     4002H    ; counter seg
c_min           EQU     4003H    ; counter min
c_hor           EQU     4004H    ; counter hor
c_dis           EQU     4005H    ; counter dis
c_dim           EQU     4006H    ; counter dim
c_mes           EQU     4007H    ; counter mes

                                 ; bloque de configuracion (0 00)
flagxc0         EQU     06H      ; banderas de config en ramx
mset0           EQU     0DH      ; memoria seteada
nom_dir         EQU     0FH      ; banco nombre del directorio
nom_dirf        EQU     16H      ; ultimo caracter del nom dir
ini_dir         EQU     17H      ; primer bloque del directorio (0 01)[1]
fin_dir         EQU     1F72H    ; ultimo bloque del directorio (1 5E)[350]


;       ********        variables
ndcnor          EQU     0AH      ; nibble de configuracion abonado normal
ndchl1          EQU     0BH      ; nibble de configuracion abonado hot para lt1
ndchl2          EQU     0CH      ; nibble de configuracion abonado hot para lt2
by_blo          EQU     17H      ; bytes por bloque
stack           EQU     4FH      ; inicio stack
car_mset        EQU     0CCH     ; caracter de memoria seteada
car_findir      EQU     5EH      ; caracter de fin de directorio
svl             EQU     0FEH     ; velocidad serial = 9600bps
careqrx         EQU     0DBH     ; caracter de control para modo recepcion
careqtx         EQU     0BDH     ; caracter de control para modo transmision


;&&     ************************************************************
;                       VECTOR DE INTERRUPCIONES
;       ************************************************************

                ORG     00H                     ; reset
                AJMP    inicio

                ORG     03H                     ; int externa 0
                AJMP    intx0

                ORG     0BH                     ; timer 0
                AJMP    intt0

                ORG     13H                     ; int externa 1
                AJMP    intx1

                ORG     1BH                     ; timer 1
                AJMP    intt1

                ORG     23H                     ; serial
                AJMP    ints
```

```
;&&     ************************************************************
;                       ATENCION A INTERRUPCIONES
;       ************************************************************

                ORG     30H


;       ------------------------------
;       externa 0
;       disparo monoestable para llin1
;       ------------------------------
intx0:          MOV     TL1,#00H                ; max ret: 106.66ms
                MOV     TH1,#00H
                MOV     cllin1,#3AH             ; 58 ciclos = 6.186666 seg
                LCALL   c_r_llin1
                SETB    TR1                     ; habilitar timer 1
                RETI


;       ------------------------------
;       timer 0
;       ------------------------------
intt0:          RETI


;       ------------------------------
;       externa 1
;       disparo monoestable para llin2
;       ------------------------------
intx1:          MOV     TL1,#00H                ; max ret: 106.66ms
                MOV     TH1,#00H
                MOV     cllin2,#3AH             ; 58 ciclos = 6.186666 seg
                LCALL   c_r_llin2
                SETB    TR1                     ; habilitar timer 1
                RETI


;       ------------------------------
;       timer 1
;       temporizacion monoestables
;       ------------------------------
intt1:          JB      r_llin1,intt1_s2        ; llin1
                DJNZ    cllin1,intt1_s2
                LCALL   s_r_llin1

intt1_s2:       JB      r_llin2,intt1_s4        ; llin2
                DJNZ    cllin2,intt1_s6
                LCALL   s_r_llin2

intt1_s4:       JNB     r_llin1,intt1_s6
                JNB     r_llin2,intt1_s6
                CLR     TR1                     ; deshabilitar timer 1
intt1_s6:       RETI


;       ------------------------------
;       serial
;       ------------------------------
ints:           RETI
```

```
;&&     ***********************************************************
;                       INICIALIZACION DEL EQUIPO
;       Establecer las caracteristicas por defecto del equipo:
;               - temporizadores para modo normal o serial,
;               - estado de los reles,
;               - display,
;               - escritor de tonos.
;       Dar paso a cualquiera de los dos modos de trabajo: SERIAL o NORMAL,
;       dependiendo de si el cable de comunicaciones se halla o no conectado.
;       ***********************************************************

inicio:
;       ********        bancos / punteros / banderas
                MOV     SP,#stack               ; direccionar stack
                MOV     R0,#20H                 ; clr banderas y flagic0: 20H - 26H
ini_s2:         MOV     @R0,#00H
                INC     R0
                CJNE    R0,#27H,ini_s2

;       ********        temporizadores, reles/filtros, carga configuracion RAMx
                JNB     ix_ser,ini_s4           ; ix_ser=1 => modo serial
                MOV     TMOD,#21H               ; t0: temp, 16 bits, habil por soft
                                                ; t1: temp, 8 bits, rec aut
                MOV     ban_rele,#0DDH          ; bloqueo lt1 y lt2
                AJMP    ini_s12

ini_s4:         MOV     TMOD,#11H               ; t0: temp 16 bits, habil por soft
                                                ; t1: temp 16 bits, habil por soft

                MOV     DPTR,#mset0             ; si la memoria no ha sido seteada,
                MOVX    A,@DPTR                 ; deja el 00H cargado al inicio
                CJNE    A,#car_mset,ini_s10
                INC     DPTR
                MOVX    A,@DPTR
                CJNE    A,#car_mset,ini_s10

                MOV     DPTR,#flagxc0           ; memoria seteada: cargar
                MOVX    A,@DPTR                 ; configuracion de ram externa
                MOV     flagic0,A

                JNB     f_bt2,ini_s10
                MOV     ban_rele,#9CH           ; bloqueo total para lt2
                AJMP    ini_s12

ini_s10:        MOV     ban_rele,#0CCH          ; no bloqueos
ini_s12:        LCALL   s_rele

;       ********        tonos wr
                MOV     DPTR,#ramx4             ; marcador ocio
                MOV     A,#0F0H
                MOVX    @DPTR,A

;       ********        inicializacion display
                LCALL   p_lcd_ini

;       ********        definir modo de trabajo
                JB      ix_ser,p_ser
                AJMP    p_nor                   ; si cable no conectado => modo normal
```

```
;&&     ************************************************************
;                       MODO SERIAL
;
;       Ingreso: el cable de comunicaciones debe hallarse conectado antes de
;                encender o resetear el equipo.
;       Salida:  resetear o encender el equipo luego de retirado el cable de
;                comunicaciones.
;       Modo de transmision: serial asincronico sin control de errores.
;       La palabra de dato: 10 bits
;               -       1 bit de inicio (0l)
;               -       8 bits de datos (LSB primero)
;               -       1 bit de parada (1l)
;       Modo de puerto: 1
;       Velocidad:      9600bps
;       Temporizador:   1 en modo 2 (autorecarga), valor de recarga 256 (0FEH)
;       SMOD:           0
;       fck:            7.3728MHz
;
;       Funciones:      TX - RX, a las funciones se ingresa mediante un caracter
;                       de control (careqrx, careqtx) enviado por el PC.
;                       En TX, si la memoria no ha sido seteada, solo transmite
;                       dos caracteres de fin de archivo
;       ************************************************************

p_ser:
;       ********        seteo parametros de comunicacion
                MOV     TL1,#svl                ; vel serial = 9600 bps
                MOV     TH1,#svl
                SETB    TR1                     ; timer 1 habilitado

                MOV     SCON,#50H               ; SM0 SM1 SM2 REN TB8 RB8 TI  RI
                                                ;  0   1   0   1   0   0   0   0
                                                ; modo 1 [SM0 SM1]
                                                ; rx habilitado [REN]
                                                ; clr banderas de rx y tx [TI, RI]

;       ********        esperando caracter de control
ser_s2:         MOV     DPTR,#men_ser
                LCALL   s_alarmas

ser_s10:        NOP
                JNB     RI,ser_s10
                CLR     RI
                MOV     A,SBUF

;       ********        recepcion del directorio
                CJNE    A,#careqrx,ser_s30
                MOV     DPTR,#men_rx
                LCALL   s_alarmas

                MOV     DPTR,#mset0             ; quitar car de mem set
                CLR     A
                MOVX    @DPTR,A
                INC     DPTR
                MOVX    @DPTR,A

ser_s20:        INC     DPTR                    ; carga nom dir y dir
ser_s22:        NOP
                JNB     RI,ser_s22
                CLR     RI
                MOV     A,SBUF
                MOVX    @DPTR,A

                CJNE    A,#car_findir,ser_s20
ser_s24:        MOVX    @DPTR,A                 ; limpiar resto de memoria
                INC     DPTR
                MOV     R4,DPH
                CJNE    R4,#20H,ser_s24         ; para 8Kx8

                MOV     DPTR,#mset0             ; poner car de mem set
                MOV     A,#car_mset
                MOVX    @DPTR,A
                INC     DPTR
                MOVX    @DPTR,A
```

```
                MOV     DPTR,#men_frx
                LCALL   s_alarmas
                AJMP    ser_s2

;       ********        transmision del directorio
ser_s30:        CJNE    A,#careqtx,ser_s10
                MOV     DPTR,#men_tx
                LCALL   s_alarmas

                MOV     DPTR,#mset0             ; revisar si la memoria se
                MOVX    A,@DPTR                 ; halla seteada
                CJNE    A,#car_mset,ser_s40
                INC     DPTR
                MOVX    A,@DPTR
                CJNE    A,#car_mset,ser_s40

                MOV     DPTR,#nom_dir           ; memoria seteada
ser_s32:        MOVX    A,@DPTR                 ; tx nom dir y dir
                MOV     SBUF,A

ser_s34:        NOP
                JNB     TI,ser_s34
                CLR     TI
                INC     DPTR
                CJNE    A,#car_findir,ser_s32

ser_s36:        MOV     DPTR,#men_ftx
                LCALL   s_alarmas
                AJMP    ser_s2

ser_s40:        MOV     R1,#02H                 ; memoria nueva
                MOV     A,#car_findir

ser_s42:        MOV     SBUF,A                  ; transmitir dos veces el
ser_s44:        NOP                             ; carfindir
                JNB     TI,ser_s44
                CLR     TI
                DJNZ    R1,ser_s42
                AJMP    ser_s36
```

```
;&&     ************************************************************
;                       MODO NORMAL
;
;       Funciones:
;       -       Limpiar automaticamente la memoria de datos si esta no se
;               encuentra seteada, sino, permite una limpieza manual (pedida
;               por el usuario mediante sw1).
;       -       Programacion de interrupciones y prioridades.
;       -       Indicar: nombre del directorio de trabajo, configuracion general
;               del equipo, lineas conmutadas directas programadas.
;       -       Barrer las lineas: si una requiere atencion, programar los
;               bloqueos sobre la otra (si los hay) y llamar a la subrutina
;               apropiada. Si ninguna requiere atencion, mostrar la hora y fecha
;               del sistema o cronometra la(s) linea(s) atendida(s).
;       ************************************************************

p_nor:
nor_s0:         LCALL   s_clr_dir               ; limpieza del directorio

;       ********        nombre del directorio de trabajo
                MOV     DPTR,#nom_dirf          ; si dir sin nombre, saltar
                MOVX    A,@DPTR                 ; esta funcion
                CJNE    A,#20H,nor_s2
                AJMP    nor_s10

nor_s2:         LCALL   s_lcd_clr
                MOV     A,#82H                  ; posicionar cursor
                LCALL   s_lcd_wrd

                MOV     DPTR,#nom_dir
                MOV     R0,#08H                 ; nombre de ocho caracteres
nor_s4:         MOVX    A,@DPTR
                INC     DPTR
                LCALL   s_lcd_wr
                DJNZ    R0,nor_s4

                MOV     DPTR,#men_ext           ; escribir extension
                LCALL   s_lcd_romy
                LCALL   s_ret_max               ; mostrar nombre por 1 seg

;       ********        configuracion
nor_s10:        LCALL   s_lcd_cnf               ; mostrar configuracion
                LCALL   s_ret_max               ; por 1 seg
                LCALL   s_lcd_hld               ; mostrar lineas hot definidas

;       ********        interrupciones & prioridades
                MOV     IP,#0DH                 ; alta pri: intx0, intx1, intt1
                                                ; - intx para saber cuando hay llin
                                                ; - timer1 por ser el usado para que
                                                ;   el uC actue como monoestable

                SETB    IT0                     ; intx0 por flanco
                CLR     IE0                     ; eliminar llamadas anteriores
                SETB    EX0                     ; inx0 habilitada

                SETB    IT1                     ; intx1 por flanco
                CLR     IE1                     ; eliminar llamadas anteriores
                SETB    EX1                     ; inx1 habilitada

                SETB    ET1                     ; intt1 habilitada
                SETB    EA                      ; habil interrupciones



;       ----------------------------
;       barrido lt1
;       ----------------------------
;                         0l                      1l
;       ix_llin         existe llin             no existe llin
;       ix_lt           linea ocupada           linea desocupada
;       f_att           linea NO atendida       linea YA atendida

nor_s20:        JB      ix_lt1,nor_s40          ; lt1 ocupada?
                JB      f_att1,nor_s60          ; lt1 ya atendida?
```

```
                JB      r_llin1,nor_s22
                SETB    f_att1                  ; si es llin no atiende
                AJMP    nor_s60

;       ********        habilitar bloqueo sobre lt2
nor_s22:        CLR     f_cro2                  ; deshabil cronometro de LT2
                JB      f_bi2,nor_s24
                JB      f_bn2,nor_s24
                AJMP    nor_s26

nor_s24:        LCALL   c_r_flt2                ; habil filtro lt2
                LCALL   s_r_lt2                 ; bloqueo en lt2

;       ********        atencion a lt1
nor_s26:        LCALL   p_lt1

;       ********        deshabilitar bloqueos sobre lt2
                MOV     R6,#0FH                 ; sensar lt2 durante 150ms para
nor_s30:        LCALL   s_ret_10                ; determinar si se halla ocupada
                JNB     ix_lt2,nor_s32          ; .or. marcando en pulsos, en
                DJNZ    R6,nor_s30              ; cualquiera de los dos casos
                AJMP    nor_s34                 ; se marca a lt2 como atendida

nor_s32:        SETB    f_att2
nor_s34:        JB      f_bi2,nor_s36
                JB      f_bn2,nor_s36
                AJMP    nor_s60

nor_s36:        LCALL   c_r_lt2                 ; quitar bloqueo en lt2
                LCALL   s_r_flt2                ; deshabil filtro lt2
                AJMP    nor_s60

;       ********        linea desocupada
nor_s40:        JNB     f_att1,nor_s60
                MOV     R6,#1EH                 ; lt1 va de ocupada a desocupada
nor_s42:        LCALL   s_ret_10
                JNB     ix_lt1,nor_s60          ; si lt1 pasa mas de 300ms en
                DJNZ    R6,nor_s42              ; alto, validar condicion de
                CLR     f_att1                  ; lt1 desocupada
                CLR     f_cro1                  ; no cronometrar lt1


;       ----------------------------
;       barrido lt2
;       ----------------------------
nor_s60:        JB      ix_lt2,nor_s80          ; lt2 ocupada?
                JB      f_att2,nor_s100         ; lt2 ya atendida?
                JB      r_llin2,nor_s62
                SETB    f_att2                  ; si es llin no atiende
                AJMP    nor_s100

;       ********        habilitar bloqueo sobre lt1
nor_s62:        CLR     f_cro1                  ; deshabil cronometro de LT1
                JB      f_bi1,nor_s64
                JB      f_bn1,nor_s64
                AJMP    nor_s66

nor_s64:        LCALL   c_r_flt1                ; habil filtro lt1
                LCALL   s_r_lt1                 ; bloqueo en lt1

;       ********        atencion a lt2
nor_s66:        LCALL   p_lt2

;       ********        deshabilitar bloqueos sobre lt1
                MOV     R6,#0FH                 ; sensar lt1 durante 150ms para
nor_s70:        LCALL   s_ret_10                ; determinar si se halla ocupada
                JNB     ix_lt1,nor_s72          ; .or. marcando en pulsos, en
                DJNZ    R6,nor_s70              ; cualquiera de los dos casos
                AJMP    nor_s74                 ; se marca a lt1 como atendida

nor_s72:        SETB    f_att1
nor_s74:        JB      f_bi1,nor_s76
                JB      f_bn1,nor_s76
```

```
                AJMP    nor_s100

nor_s76:        LCALL   c_r_lt1                 ; quitar bloqueo en lt1
                LCALL   s_r_flt1                ; deshabil filtro lt1
                AJMP    nor_s100


;       ********        linea desocupada
nor_s80:        JNB     f_att2,nor_s100
                MOV     R6,#1EH                 ; lt2 va de ocupada a desocupada
nor_s82:        LCALL   s_ret_10
                JNB     ix_lt2,nor_s100         ; si lt2 pasa mas de 300ms en
                DJNZ    R6,nor_s82              ; alto, validar condicion de
                CLR     f_att2                  ; lt2 desocupada
                CLR     f_cro2                  ; no cronometrar lt2


;       ----------------------------
;       reloj - cronometro
;
;       f_cro1=0 .AND. f_cro2=0  => fecha y hora
;       f_cro1=1 .OR.  f_cro2=0  => cronometro
;       Cronometradas llamadas hechas a traves del equipo
;       ----------------------------
nor_s100:       JB      f_cro1,nor_s102
                JB      f_cro2,nor_s102
                LCALL   s_feho
                LCALL   s_ret_100               ; barrido sobre las lineas 10Hz
                AJMP    nor_s20

nor_s102:       LCALL   s_crono
                LCALL   s_ret_100               ; barrido sobre las lineas 10Hz
                AJMP    nor_s20
```

```
;&&     ************************************************************
;                       MANEJO DE LAS LINEAS
;       -       Atencion a las lineas
;       -       Equipo activo
;       -       Modo directorio
;       -       Modo marcacion abreviada
;       ************************************************************

;       ##############################
;       p_lt1
;       procedimiento atencion a lt1
;       teclado A
;
;       funciones:
;       -       Si abonado HOT, marca el numero; para inhibir el marcado
;               automatico, antes de ocupar la linea pulsar el sw1
;       -       Si el equipo es normalmente activo, poner telefono en estado
;               polarizado y entra al menu principal.
;       -       Si el equipo es normalmente apagado, censar hasta las tres
;               primeras teclas presionadas con el fin de determinar:
;                       codigo de activar equipo:       0*
;                       ejecutada ldi:                  116 / 00
;                       ejecutada ldn:                  105 / 1=x<=8
;               si hay bloqueos, evita la ejecucion de la llamada; sino, lee el
;               numero marcado hasta detectar los 425Hz (si es en tonos) o
;               hasta despues de 12seg de marcado el ultimo digito (si es en
;               pulsos) y da la orden de cronometrar la llamada.
;       -       Determinar automaticamente si el telefono opera en modo tonos
;               o modo pulsos.
;       -       Si hay peticion de marcado automatico (numero localizado en el
;               directorio) marcar ssi no hay llin.
;       ##############################

p_lt1:          CLR     f_lt2att
                CLR     f_cndc
                LCALL   s_fhl1                  ; lt1 es hot?
                JNB     f_fndc,lt1_s2

;       ********        lt1 es conmutada directa
                JB      ix_sw1,lt1_s2           ; inhibir llamada hot
                LCALL   s_lcd_ram               ; mostrar
                LCALL   s_ret_1000
                LCALL   s_mar1                  ; marcar
                RET

;       ----------------------------
;       lt1 de tonos
;       ----------------------------
lt1_s2:         JB      f_ne,lt1_s22            ; f_ne=1 => equipo normalmente
                                                ; encendido
lt1_s10:        MOV     DPTR,#men_lt1
                LCALL   s_lcd_rom

                CLR     f_on
                CLR     f_ldi
                CLR     f_ldn
                CLR     f_cel
                CLR     f_k1
                MOV     R0,#03H                 ; R0 contador de las tecla presionadas
                LCALL   s_r_dr1                 ; conexion lector dtmf

;       ********'       revision tres primeras teclas presionadas
lt1_s12:        NOP
                JNB     ix_lt1,lt1_s14          ; lt1 sigue ocupada?
                AJMP    lt1_s70
lt1_s14:        JNB     ix_dtmf,lt1_s12         ; existe tecla presionada?

                LCALL   s_lcd_wrtx              ; teclado A: externo
                LCALL . s_tpn

lt1_s16:        NOP                             ; espera tecla liberada
                JB      ix_dtmf,lt1_s16
                JB      f_k1,lt1_s20
                DJNZ    R0,lt1_s12
```

```
;       ********        equipo activo
lt1_s20:        JNB     f_on,lt1_s30            ; codigo de activado recibido
lt1_s22:        LCALL   s_r_lt1                 ; desconexion telf-lt (telf pol)
                LCALL   s_r_dr1                 ; conexion r_dr para cuando eqNE
                LCALL   p_eqac

                JB      r_llin1,lt1_s24         ; r_llin1=1 => NO existe llin1
                LCALL   c_r_dr1                 ; evitar que senal de timbre llegue al
                LCALL   c_r_lt1                 ; secundario del transformador
                SETB    f_att1
                MOV     DPTR,#men_llin1         ; atender llin
                LCALL   s_alarma
                RET

lt1_s24:        LCALL   c_r_lt1                 ; conexion telf-lt (telf normal)
                JB      f_na,lt1_s10
                JNB     f_mar,lt1_s26
                LCALL   s_mar1                  ; marcar el numero hallado en el banco

lt1_s26:        LCALL   c_r_dr1                 ; desconexion lector dtmf
                RET

;       ********        revision llamadas prohibidas
lt1_s30:        MOV     A,#85H                  ; posicionar cursor

                JNB     f_ldn,lt1_s32           ; ldn
                MOV     DPTR,#men_ldn
                LCALL   s_lcd_romx
                JB      f_bn1,lt1_s60
                AJMP    lt1_s40

lt1_s32:        JNB     f_ldi,lt1_s34           ; ldi
                MOV     DPTR,#men_ldi
                LCALL   s_lcd_romx
                JB      f_bi1,lt1_s60
                AJMP    lt1_s40

lt1_s34:        JNB     f_cel,lt1_s36           ; llamada a celular
                MOV     DPTR,#men_cel
                LCALL   s_lcd_romx
                AJMP    lt1_s40

lt1_s36:        MOV     DPTR,#men_loc           ; llamada local
                LCALL   s_lcd_romx

;       ********        llamada autorizada
lt1_s40:        MOV     DPTR,#men_ok
                LCALL   s_lcd_romy

lt1_s42:        MOV     R0,#ss1                 ; marcado el num completo
                LCALL   s_cha_hex               ; carga la hora de inicio de
                SETB    f_cro1                  ; la llout1
                SETB    f_att1                  ; lt1 ya atendida
                LCALL   c_r_dr1                 ; desconexion lector dtmf
                LCALL   s_ret_500
                RET

;       ********        bloqueo
lt1_s60:        LCALL   s_r_lt1                 ; llamada SI es prohibida
                MOV     A,#0C7H                 ; mensaje bajo tipo de llamada
                MOV     DPTR,#men_blo
                LCALL   s_alarmax

lt1_s62:        NOP                             ; bloqueo mientras descolgado
                JB      ix_tel1,lt1_s62         ; ix_tel1=1 => telf descolgado

                MOV     R6,#00H                 ; validar telefono colgado
lt1_s64:        LCALL   s_ret_10                ; si este pasa asi por mas de
                INC     R6                      ; 300ms
                JB      ix_tel1,lt1_s62
                CJNE    R6,#1EH,lt1_s64
                LCALL   c_r_lt1
                RET
```

```
;       ------------------------------
;       lt1 de pulsos
;       ------------------------------
;       Si en el lazo lt1_s12 ix_lt1=1 (linea desocupada), se revisa si se
;       trata de una marcacion en pulsos viendo si se presenta una cadencia
;       de altos y bajos.
;       Si un alto se presenta mas de 300ms, se considera linea desocupada
;       Cadencia:       alto    60ms
;                       bajo    40ms
;       Tiempo de validacion de un numero: 500ms

lt1_s70:        LCALL   c_r_dr1

lt1_s72:        LCALL   s_rd_pls
                JNB     ix_lt1,lt1_s74
                RET

lt1_s74:        LCALL   s_tpn
                JNB     f_k1,lt1_s76
                AJMP    lt1_s30
lt1_s76:        DEC     R0

lt1_s78:        NOP                             ; esperar siguiente numero
                JNB     ix_lt1,lt1_s78
                AJMP    lt1_s72

;       ********        lectura resto del numero pulsos
lt1_s80:        MOV     R3,#00H
lt1_s82:        MOV     R2,#00H

lt1_s84:        LCALL   s_ret_10                ; temporizar 12 seg para
                INC     R2                      ; validar fin de numero marcado
                JB      ix_lt1,lt1_s86
                CJNE    R2,#78H,lt1_s84
                INC     R3
                CJNE    R3,#0AH,lt1_s82
                AJMP    lt1_s42

lt1_s86:        LCALL   s_rd_pls
                JNB     ix_lt1,lt1_s80
                RET



;&&     ##############################
;       p_lt2
;       atencion a lt2
;       ##############################

p_lt2:          SETB    f_lt2att
                CLR     f_cndc
                LCALL   s_fhl2                  ; lt2 es hot?
                JNB     f_fndc,lt2_s2

;       ********        lt2 es conmutada directa
                JB      ix_sw1,lt2_s2           ; inhibir llamada hot
                LCALL   s_lcd_ram               ; mostrar
                LCALL   s_ret_1000
                LCALL   s_mar2                  ; marcar
                RET

;       ------------------------------
;       lt2 de tonos
;       ------------------------------
lt2_s2:         JB      f_ne,lt2_s22            ; f_ne=1 => equipo normalmente
                                                ; encendido
lt2_s10:        MOV     DPTR,#men_lt2
                LCALL   s_lcd_rom

                CLR     f_on
                CLR     f_ldi
                CLR     f_ldn
```

```
                CLR     f_cel
                CLR     f_k1
                MOV     R0,#03H                 ; R0 contador de las tecla presionadas
                LCALL   s_r_dr2                 ; conexion lector dtmf

;       ********        revision tres primeras teclas presionadas
lt2_s12:        NOP
                JNB     ix_lt2,lt2_s14          ; lt2 sigue ocupada?
                AJMP    lt2_s70
lt2_s14:        JNB     ix_dtmf,lt2_s12         ; existe tecla presionada?

                LCALL   s_lcd_wrtx              ; teclado A: externo
                LCALL   s_tpn

lt2_s16:        NOP                             ; espera tecla liberada
                JB      ix_dtmf,lt2_s16

                JB      f_k1,lt2_s20
                DJNZ    R0,lt2_s12

;       ********        equipo activo
lt2_s20:        JNB     f_on,lt2_s30            ; codigo de activado recibido
lt2_s22:        LCALL   s_r_lt2                 ; desconexion telf-lt (telf pol)
                LCALL   s_r_dr2                 ; conexion r_dr para cuando eqNE
                LCALL   p_eqac

                JB      r_llin2,lt2_s24         ; r_llin2=1 => NO existe llin1
                LCALL   c_r_dr2                 ; evitar que senal de timbre llegue al
                LCALL   c_r_lt2                 ; secundario del transformador
                SETB    f_att2
                MOV     DPTR,#men_llin2         ; atender llin
                LCALL   s_alarma
                RET

lt2_s24:        LCALL   c_r_lt2                 ; conexion telf-lt (telf normal)
                JB      f_na,lt2_s10
                JNB     f_mar,lt2_s26
                LCALL   s_mar2                  ; marcar el numero hallado en el banco

lt2_s26:        LCALL   c_r_dr2                 ; desconexion lector dtmf
                RET

;       ********        revision llamadas prohibidas
lt2_s30:        MOV     A,#85H                  ; posicionar cursor

                JNB     f_ldn,lt2_s32           ; ldn
                MOV     DPTR,#men_ldn
                LCALL   s_lcd_romx
                JB      f_bn2,lt2_s60
                AJMP    lt2_s40

lt2_s32:        JNB     f_ldi,lt2_s34           ; ldi
                MOV     DPTR,#men_ldi
                LCALL   s_lcd_romx
                JB      f_bi2,lt2_s60
                AJMP    lt2_s40

lt2_s34:        JNB     f_cel,lt2_s36           ; llamada a celular
                MOV     DPTR,#men_cel
                LCALL   s_lcd_romx
                AJMP    lt2_s40

lt2_s36:        MOV     DPTR,#men_loc           ; llamada local
                LCALL   s_lcd_romx

;       ********        llamada autorizada
lt2_s40:        MOV     DPTR,#men_ok
                LCALL   s_lcd_romy

lt2_s42:        MOV     R0,#ss2                 ; marcado el num completo
                LCALL   s_cha_hex               ; carga la hora de inicio de
                SETB    f_cro2                  ; la llout2
                SETB    f_att2                  ; lt2 ya atendida
                LCALL   c_r_dr2                 ; desconexion lector dtmf
```

```
                LCALL   s_ret_500
                RET

;       ********        bloqueo
lt2_s60:        LCALL   s_r_lt2                 ; llamada SI es prohibida
                MOV     A,#0C7H                 ; mensaje bajo tipo de llamada
                MOV     DPTR,#men_blo
                LCALL   s_alarmax

lt2_s62:        NOP                             ; bloqueo mientras descolgado
                JB      ix_tel2,lt2_s62         ; ix_tel1=1 => telf descolgado

                MOV     R6,#00H                 ; validar telefono colgado
lt2_s64:        LCALL   s_ret_10                ; si este pasa asi por mas de
                INC     R6                      ; 300ms
                JB      ix_tel2,lt2_s62
                CJNE    R6,#1EH,lt2_s64

                LCALL   c_r_lt2
                RET

;       -----------------------------
;       lt2 de pulsos
;       -----------------------------
lt2_s70:        LCALL   c_r_dr2

lt2_s72:        LCALL   s_rd_pls
                JNB     ix_lt2,lt2_s74
                RET

lt2_s74:        LCALL   s_tpn
                JNB     f_k1,lt2_s76
                AJMP    lt2_s30
lt2_s76:        DEC     R0

lt2_s78:        NOP                             ; esperar siguiente numero
                JNB     ix_lt2,lt2_s78
                AJMP    lt2_s72

;       ********        lectura resto del numero pulsos
lt2_s80:        MOV     R3,#00H
lt2_s82:        MOV     R2,#00H

lt2_s84:        LCALL   s_ret_10
                INC     R2
                JB      ix_lt2,lt2_s86
                CJNE    R2,#78H,lt2_s84
                INC     R3
                CJNE    R3,#0AH,lt2_s82
                AJMP    lt2_s42

lt2_s86:        LCALL   s_rd_pls
                JNB     ix_lt2,lt2_s80
                RET
```

```
;&&     ==============================
;       p_eqac
;       procedimiento equipo activo
;       teclado B
;
;       funciones:
;       [*]     regreso a modo normalmente apagado
;       [#]     SOLO PARA lt1, ingreso a modo programacion ssi la llave se halla
;               colocada, sino, aparece el mensaje de informacion protegida.
;       [0]     ingreso a marcacion abreviada (ssi hay abonados en el directorio).
;       [2] - [9]       ingreso a directorio (ssi hay abonados en el directorio).
;
;       OUT     f_na, f_mar
;       ==============================

p_eqac:         CLR     f_na                    ; regreso a mormalmente apagodo NO
                CLR     f_mar                   ; marcar NO
eqac_s2:        MOV     DPTR,#men_eqac
                LCALL   s_lcd_rom

eqac_s4:        LCALL   s_tecla1                ; teclado B: equipo activo
                JB      f_lt2att,eqac_s6
                JB      ix_tel1,eqac_s10
                RET

eqac_s6:        JB      ix_tel2,eqac_s10
                RET

eqac_s10:       CJNE    A,#20H,eqac_s12         ; [1] => no funcion
                LCALL   s_beep
                AJMP    eqac_s2

eqac_s12:       CJNE    A,#2AH,eqac_s14         ; [*] regresar a NA
                SETB    f_na
                RET

eqac_s14:       CJNE    A,#23H,eqac_s20         ; [#] programacion
                JB      f_lt2att,eqac_s18       ; ingreso con lt1 only
                JNB     ix_lv,eqac_s16
                LCALL   p_prog                  ; inf no protegida: llave puesta
                AJMP    eqac_s2

eqac_s16:       MOV     DPTR,#men_infp          ; inf protegida: llave NO puesta
                LCALL   s_alarma
                AJMP    eqac_s2

eqac_s18:       MOV     DPTR,#men_prplt1        ; programar solo desde lt1
                LCALL   s_alarma
                AJMP    eqac_s2

eqac_s20:       PUSH    A                       ; si directorio vacio no
                MOV     DPTR,#ini_dir           ; entra a modo directorio
                MOVX    A,@DPTR                 ; ni marc abrev
                CJNE    A,#car_findir,eqac_s22
                POP     A
                MOV     DPTR,#men_dirvac        ; directorio vacio
                LCALL   s_alarma
                AJMP    eqac_s2

eqac_s22:       POP     A
                CJNE    A,#30H,s_dir            ; [0] marc abrev
                AJMP    s_abr
```

```
;&&     ------------------------------
;       s_dir
;       directorio
;       teclado C
;
;       funciones:
;       - Localizar el primer abonado del grupo definido por la tecla:
;               [2]     grupos A, B, C
;               [3]     grupos D, E, F
;               [4]     grupos G, H, I
;               [5]     grupos J, K, L
;               [6]     grupos M, N, O
;               [7]     grupos P, Q, R, S
;               [8]     grupos T, U, V
;               [9]     grupos W, X, Y, Z
;         Si no hay abonados en el grupo, se indica mediante un mensaje.
;       - Navegar por el directorio:
;               [1]     ir al anterior abonado
;               [0]     ir al siguiente abonado
;         Indica inicio o fin del directorio (BOF), (EOF).
;       - [#] set bandera f_mar (si numero en pantalla no tiene bloqueo) para
;           que p_lt.. marque
;       - [*] set bandera f_na para que p_lt.. regrese al menu principal.
;
;       OUT     f_mar, f_na
;       ------------------------------

s_dir:
dir_s0:         MOV     tec_pre,A               ; directorio
                CLR     f_tpos
                MOV     R3,#00H                 ; R3, contador de las veces

dir_s2:         LCALL   s_find
                JNB     f_nf,dir_s4             ; grupo sin abonados
                LCALL   s_lcd_nf
                AJMP    dir_s6

dir_s4:         LCALL   s_lcd_ram
dir_s6:         LCALL   s_tecla0                ; teclado C: directorio
                JB      f_lt2att,dir_s8
                JB      ix_tel1,dir_s10
                RET

dir_s8:         JB      ix_tel2,dir_s10
                RET

dir_s10:        CJNE    A,#2AH,dir_s20          ; [*] salir
                AJMP    eqac_s2

dir_s20:        CJNE    A,#23H,dir_s30          ; [#] marcar
                MOV     A,numero
                JZ      dir_s6                  ; no marca si grupo sin abonados

                JB      f_lt2att,dir_s22
                JB      f_bi1,dir_s26           ; bloqueos para lt1
                JB      f_bn1,dir_s26
                AJMP    dir_s24

dir_s22:        JB      f_bi2,dir_s26           ; bloqueos para lt2
                JB      f_bn2,dir_s26

dir_s24:        SETB    f_mar
                RET

dir_s26:        LCALL   s_tpni
                JNB     f_tpni,dir_s24          ; f_tpni=1 => numero con bloqueo
                AJMP    dir_s4

dir_s30:        LCALL   s_t_alfn                ; teclado alfabetico y de naveg
                JNB     f_alf,dir_s4            ; f_alf=0 => navegacion
                AJMP    dir_s2
```

```
;&&     ------------------------------
;       s_abr
;       marcacion abreviada
;       teclado D
;
;       funciones:
;       -  Indicar que abreviados se hallan asignados, si no hay asignados,
;          sale automaticamente.
;       -  [0]...[9]    marcar el numero del abonado abreviado indicado por
;                       la tecla presionada, si esta asignado y si la llamada
;                       no tiene bloqueo.
;       -  [*]          regresar al menu de ATENCION A LA LINEA.
;       ------------------------------

s_abr:
abr_s0:         LCALL   s_lcd_abra              ; indicar abreviados asignados
                JNB     f_eaua,abr_s14

abr_s2:         LCALL   s_tecla5                ; teclado D: marc abrev
                JB      f_lt2att,abr_s4
                JB      ix_tel1,abr_s6
                RET

abr_s4:         JB      ix_tel2,abr_s6
                RET

abr_s6:         CJNE    A,#20H,abr_s10          ; [*] salir
                AJMP    eqac_s2

abr_s10:        CJNE    A,#3AH,abr_s12          ; [#] sin funcion
                MOV     A,#0C2H                 ; limpiar caracter escrito
                LCALL   s_lcd_wrd
                MOV     A,#5FH                  ; "_"
                LCALL   s_lcd_wr
                MOV     A,#0C2H
                LCALL   s_lcd_wrd
                LCALL   s_beep
                AJMP    abr_s2

abr_s12:        ANL     A,#0FH
                LCALL   s_fabr
                JB      f_fndc, abr_s20
abr_s14:        MOV     DPTR,#men_abrna
                LCALL   s_alarma
                AJMP    eqac_s2

abr_s20:        LCALL   s_lcd_ram
                JB      f_lt2att,abr_s22
                JB      f_bi1,abr_s30           ; bloqueos para lt1
                JB      f_bn1,abr_s30
                AJMP    abr_s24

abr_s22:        JB      f_bi2,abr_s30           ; bloqueos para lt2
                JB      f_bn2,abr_s30

abr_s24:        SETB    f_mar
                RET

abr_s30:        LCALL   s_tpni
                JNB     f_tpni,abr_s24
                AJMP    eqac_s2
```

```
;&&     ************************************************************
;                       PROGRAMACION
;       -       menu principal
;       -       agregar abonado
;       -       eliminar / definir como HOT / definir como abreviado
;       -       configuracion lineas y equipo
;       -       igualar fecha y hora del sistema
;       ************************************************************


;       ==============================
;       p_prog
;       modo programacion
;       teclado E
;
;       funciones:
;               [*]  salir a modo equipo activo
;               [#]  ejecutar seleccion
;               [7]  definicion de password DESHABILITADO
;       Directorio:
;               [2] agregar un abonado; ingresa ssi hay espacio disponible en el
;                   directorio, si no, indica que este se encuentra lleno.
;               [3] eliminar un abonado; ingresa ssi hay por lo menos un abonado
;                   en el directorio, si no, indica que este se halla vacio.
;               [4] definir a un abonado del directorio como HOT para una de las
;                   lineas; ingresa ssi hay por lo menos un abonado en el
;                   directorio, si no, indica que este se halla vacio.
;               [6] definir a un abonado del directorio como ABREVIADO para ambas
;                   lineas; ingresa ssi hay por lo menos un abonado en el
;                   directorio, si no, indica que este se halla vacio.
;       Configuracion de la linea y el equipo
;               [5] definir parametros de las lineas: tipo (tonos-pulsos),
;                   bloqueos (total solo para lt2, ldi, ldn), equipo (NA o NE).
;       Tiempo de equipo
;               [8] cambio del tiempo (fecha y hora) del equipo.
;       ==============================

p_prog:
prog_s0:        MOV     DPTR,#men_prog
                LCALL   s_lcd_rom
                MOV     A,#0CH                  ; cursor off, blink off
                LCALL   s_lcd_wrd

                MOV     R0,#17H                 ; salto por default
prog_s2:        LCALL   s_tecla3                ; teclado E: menu programacion
                JB      ix_tel1,prog_s4
                RET

prog_s4:        CJNE    A,#2AH,prog_s10         ; [*] salir
                RET

prog_s10:       CJNE    A,#41H,prog_s20         ; [2] agregar
                MOV     R0,#00H
                MOV     DPTR,#men_agre
                LCALL   s_lcd_romy

                MOV     DPTR,#fin_dir           ; revisar si el ultimo bloque
                MOVX    A,@DPTR                 ; esta o no vacio
                CJNE    A,#car_findir,prog_s12
                AJMP    prog_s2

prog_s12:       MOV     A,#0C0H                 ; cursor a la 2da fila
                MOV     DPTR,#men_dirllep       ; no existe espacio
                LCALL   s_alarmax
                AJMP    prog_s0

prog_s20:       CJNE    A,#45H,prog_s30         ; [3] eliminar
                SETB    f_elim
                CLR     f_hotl
                MOV     DPTR,#men_elim
                AJMP    prog_s100

prog_s30:       CJNE    A,#48H,prog_s40         ; [4] hot line
                CLR     f_elim
```

```
                SETB    f_hotl
                MOV     DPTR,#men_hotl
                AJMP    prog_s100

prog_s40:       CJNE    A,#4CH,prog_s50         ; [5] linea - equipo
                MOV     R0,#0AH
                MOV     DPTR,#men_line
prog_s42:       LCALL   s_lcd_romy
prog_s44:       AJMP    prog_s2

prog_s50:       CJNE    A,#4DH,prog_s60         ; [6] marcacion abreviada
                CLR     f_elim
                CLR     f_hotl
                MOV     DPTR,#men_mabr
                AJMP    prog_s100

prog_s60:       CJNE    A,#50H,prog_s70         ; [7]
                AJMP    prog_s90

prog_s70:       CJNE    A,#54H,prog_s80         ; [8] tiempo
                MOV     R0,#14H
                MOV     DPTR,#men_tiem
                AJMP    prog_s42

prog_s80:       CJNE    A,#0CEH,prog_s90        ; [#] confirmar
                MOV     DPTR,#menu_prog
                MOV     A,R0
                JMP     @A+DPTR

prog_s90:       LCALL   s_beep                  ; teclas sin funcion
                AJMP    prog_s0


;       ********        ejecutar seleccion
                                                ; DPTR +
menu_prog:      LCALL   s_agre                  ; 00H: agregar nuevo abonado
                AJMP    prog_s0

;       f_elim  f_hotl  funcion de s_ehm
;       1       0       eliminar abonado seleccionado
;       0       1       definir como hot al abonado seleccionado
;       0       0       definir como abreviado al abonado seleccionado
                LCALL   s_ehm                   ; 05H: eliminar/hot/mabr
                AJMP    prog_s0

                LCALL   s_line                  ; 0AH: linea-bloqueos / equipo
                AJMP    prog_s0

                NOP
                NOP
                NOP
                AJMP    prog_s0

                LCALL   s_tiem                  ; 14H: tiempo igualar reloj
                AJMP    prog_s0                 ; 17H

prog_s100:      MOV     R0,#05H
                LCALL   s_lcd_romy
                MOV     DPTR,#ini_dir           ; si dir vacio, no entra a s_ehm
                MOVX    A,@DPTR
                CJNE    A,#car_findir,prog_s44
                MOV     A,#0C0H                 ; cursor a la 2da fila
                MOV     DPTR,#men_dirvacp
                LCALL   s_alarmax
                AJMP    prog_s0
```

```
;&&     ==============================
;       s_agre
;       agregar un abonado
;
;       funciones:
;       1.- Permitir el ingreso del nombre y numero telefonico de un abonado.
;             - nombre:
;               . teclado F
;               . 16 caracteres (A...Z + espacio en blanco)
;               . [*] backspace
;               . [#] siguiente / espacio blanco
;               . una vez ingresado el decimo sexto caracter automaticamente
;                 pasa al ingreso del numero
;             - numero:
;               . teclado G
;               . 13 digitos (0...9)
;               . [*] backspace para corregir el ultimo digito ingresado, si no
;                     se ha ingresado ninguno, regresa al ultimo caracter del
;                     nombre
;               . [#] indicar fin del numero (numero < 14 digitos)
;               . si el numero consta de 14 digitos, al ingresar el decimo
;                 cuarto automaticamente concluye el ingreso
;       2.- teclado H
;             Una vez concluido el ingreso del numero es posible:
;             [*]     backspace, regresa al ingreso del numero
;             [0]     clear all, ingresar todo nuevamente (nombre y numero)
;             [#]     confirmar ingreso, cargar datos a la memoria ssi el
;                     banco del numero NO esta vacio
;       3.- Confirmado el ingreso, localiza el sitio del directorio donde
;           alfabeticamente tiene que ir el nuevo abonado, desplaza todos los
;           abonado que hay a partir de ahi un bloque hacia abajo y, en el
;           bloque desocupado, guarda los datos ingresados.
;           - el ordenamiento es alfabetico por el nombre del abonado
;           - si un mismo nombre es ingresado dos veces (abonado con dos
;             numeros telefonicos), el ultimo numero ingresado aparecera
;             primero en el directorio
;       4.- La unica forma de salir de la subrutina sin agregar un abonado es
;           colgando el telefono
;       ==============================

s_agre:         LCALL   s_lcd_clr               ; clr display
                MOV     R0,#nombre
agre_s2:        MOV     @R0,#20H                ; clr banco nombre (30H-3FH)
                INC     R0                      ; 20H ( ) espacio blanco
                CJNE    R0,#numero,agre_s2
agre_s4:        MOV     @R0,#3AH                ; clr banco numero (40H-4DH)
                INC     R0                      ; 4EH (:) byte de seguridad
                CJNE    R0,#numero+0FH,agre_s4

                MOV     A,#0EH                  ; cursor on, blink off
                LCALL   s_lcd_wrd

                MOV     R0,#nombre              ; puntero de los bancos
                CLR     f_tpos                  ; tecla con 3 posibilidades
agre_s6:        MOV     tec_pre,#00H            ; reset tecla pres
                MOV     R3,#00H                 ; reset contador de veces
                MOV     A,#80H                  ; cursor al inicio de la fila 2
                LCALL   s_lcd_wrd
                LCALL   s_beep

;       ********        ingreso nombre
agre_s10:       CJNE    R0,#numero,agre_s12     ; 16 caracteres
                AJMP    agre_s50

agre_s12:       LCALL   s_tecla0                ; teclado F: agregar nombre
                JB      ix_tel1,agre_s14
                RET

agre_s14:       CJNE    A,#20H,agre_s16         ; [1] tecla sin funcion
                LCALL   s_beep
                AJMP    agre_s12
```

```
agre_s16:       CJNE    A,#30H,agre_s20         ; [0] tecla sin funcion
                LCALL   s_beep
                AJMP    agre_s12

agre_s20:       CJNE    A,#2AH,agre_s40         ; [*] backspace
                CJNE    R0,#nombre+01H,agre_s22
                MOV     tec_pre,#00H            ; reset tecla pres
                AJMP    agre_s24

agre_s22:       MOV     tec_pre,#01H            ; set tecla pres
agre_s24:       MOV     A,#20H                  ; limpiar donde esta el cursor
                MOV     @R0,A
                LCALL   s_lcd_wr
                JC      agre_s6

agre_s30:       DEC     R0                      ; pone espacio en blanco en la
                MOV     A,#50H                  ; localidad anterior y ahi
                ADD     A,R0                    ; deja el cursor
                LCALL   s_lcd_wrd
                MOV     A,#20H
agre_s32:       MOV     @R0,A
                LCALL   s_lcd_wr
agre_s34:       MOV     A,#50H
                ADD     A,R0
                LCALL   s_lcd_wrd
                AJMP    agre_s10

agre_s40:       CJNE    A,#23H,agre_s42         ; [#] espacio blanco / siguiente
                MOV     A,tec_pre
                JZ      agre_s12                ; no adelanta si no ingresado
                                                ; primer caracter
                INC     R0
                MOV     tec_pre,#01H            ; set tecla pres
                AJMP    agre_s34

agre_s42:       LCALL   s_t_alf                 ; teclado alfabetico
                AJMP    agre_s32

;       ********        ingreso numero
agre_s50:       MOV     A,#0C0H                 ; cursor al inicio de la linea 2
                LCALL   s_lcd_wrd

agre_s52:       CJNE    R0,#numero+0EH,agre_s54         ; 14 numeros
                AJMP    agre_s72

agre_s54:       LCALL   s_tecla5                ; teclado G: agregar numero
                JB      ix_tel1,agre_s60
                RET

agre_s60:       CJNE    A,#20H,agre_s70         ; [*] backspace
                CJNE    R0,#numero+01H,agre_s62
agre_s62:       JC      agre_s30                ; si no hay numeros ingresados
                DEC     R0                      ; regresa al ultimo caracter
agre_s64:       MOV     A,#80H                  ; del nombre
                ADD     A,R0
                PUSH    A
                LCALL   s_lcd_wrd
                MOV     @R0,#3AH
                MOV     A,#20H
                LCALL   s_lcd_wr
                POP     A
                LCALL   s_lcd_wrd
                AJMP    agre_s54

agre_s66:       MOV     @R0,A
                INC     R0
                AJMP    agre_s52

;       ********        ingreso concluido
agre_s70:       CJNE    A,#3AH,agre_s66         ; [#] fin del numero
                CJNE    R0,#numero,agre_s74     ; si no hay numeros ingresados,
                LCALL   s_beep                  ; no carga
                AJMP    agre_s64
```

```
agre_s72:       MOV     A,#3AH                  ; ":" fin de numero
                LCALL   s_lcd_wr

agre_s74:       LCALL   s_beep
                LCALL   s_tecla0                ; teclado H: ingreso concluido
                JB      ix_tel1,agre_s80
                RET

agre_s80:       CJNE    A,#2AH,agre_s82         ; [*] backspace
                MOV     A,#80H
                DEC     R0
                ADD     A,R0
                PUSH    A
                LCALL   s_lcd_wrd
                MOV     A,#20H
                LCALL   s_lcd_wr
                LCALL   s_lcd_wr
                POP     A
                LCALL   s_lcd_wrd
                AJMP    agre_s54

agre_s82:       CJNE    A,#30H,agre_s90         ; [0] clear all
                MOV     DPTR,#men_clrall        ; ingresar nuevamente
                LCALL   s_lcd_rom
                LCALL   s_ret_500
                AJMP    s_agre

;       ********        busqueda del sitio a insertar
;       si no hay abonados en el grupo (f_nf=1), agregar en el bloque apuntado,
;       si si, restar el abonado a agregar del abonado apuntado (directorio):
;               directorio - agregado
;               si C = 0        =>      directorio > agregado
;                                       insertar en el sitio apuntado
;               si C = 1        =>      directorio < agregado
;                                       comparar con el siguiente
agre_s90:       CJNE    A,#23H,agre_s74         ; [#] carga en el directorio
                MOV     A,nombre
                LCALL   s_find
                JB      f_nf,agre_s100

agre_s92:       LCALL   s_dec_dptr              ; DEC DPTR
                MOV     stk_dph,DPH             ; salvar DPTR
                MOV     stk_dpl,DPL

                MOV     R0,#numero              ; final del banco del nombre+1
                MOV     R2,#10H                 ; 16 caracteres
                CLR     C                       ; resta sin debo inicial

agre_s94:       MOV     DPH,stk_dph
                MOV     DPL,stk_dpl
                DEC     R0

                MOV     R3,R2                   ; mandar DPTR al LSB siguiente
agre_s96:       INC     DPTR
                DJNZ    R3,agre_s96

                MOVX    A,@DPTR
                SUBB    A,@R0
                DJNZ    R2,agre_s94

                JNC     agre_s100
                LCALL   s_inc_dp                ; comparar con el sig abonado
                LCALL   s_add_ab
                AJMP    agre_s92

;       ********        desplazar abonados
agre_s100:      MOV     dir_punbk,dir_pun       ; salvar direccion del bloque
                MOV     C,f_dp                  ; donde agregar
                MOV     f_dpbk,C

agre_s102:      MOV     A,#car_findir           ; buscar fin de directorio
                LCALL   s_find
                MOV     A,dir_pun               ; ? agregar en bloque actual
                CJNE    A,dir_punbk,agre_s106
```

```
                JB      f_dpbk,agre_s104
                JB      f_dp,agre_s106
                AJMP    agre_s120               ; agregar en bloque actual!
agre_s104:      JB      f_dp,agre_s120

agre_s106:      MOV     stk_dph,DPH             ; bloque destino (siguiente)
                MOV     stk_dpl,DPL             ; guardar direccion bloque destino

                LCALL   s_dec_dp                ; bloque origen (anterior)
                LCALL   s_add_ab

                MOV     R1,#by_blo
agre_s110:      MOVX    A,@DPTR                 ; limpiar bloque origen
                PUSH    A
                MOV     A,#car_findir
                MOVX    @DPTR,A
                POP     A
                INC     DPTR
                PUSH    DPH                     ; guardar direccion bloque origen
                PUSH    DPL

                MOV     DPH,stk_dph             ; recuperar direccion bloque destino
                MOV     DPL,stk_dpl
                MOVX    @DPTR,A                 ; desplazar datos al bloque destino
                INC     DPTR
                MOV     stk_dph,DPH             ; guardar direccion bloque destino
                MOV     stk_dpl,DPL

                POP     DPL                     ; recuperar direccion bloque origen
                POP     DPH
                DJNZ    R1,agre_s110
                AJMP    agre_s102

;       ********        insertar abonado agregado
agre_s120:      MOV     R0,#nombre
agre_s122:      MOV     A,@R0                   ; cargar nombre
                MOVX    @DPTR,A
                INC     DPTR
                INC     R0
                CJNE    R0,#numero,agre_s122

                MOV     R1,#07H                 ; cargar numero
                CJNE    @R0,#30H,agre_s126      ; revisar si es ddi
                INC     R0
                CJNE    @R0,#30H,agre_s124
                MOV     @R0,#3AH                ; numero es ddi
                AJMP    agre_s126

agre_s124:      DEC     R0
agre_s126:      MOV     A,@R0
                ANL     A,#0FH
                SWAP    A

                CJNE    R1,#01H,agre_s128
                MOV     B,#0AH                  ; abonado normal (por default)
                AJMP    agre_s130

agre_s128:      INC     R0
                MOV     B,@R0
                ANL     B,#0FH
                INC     R0
agre_s130:      ORL     A,B
                MOVX    @DPTR,A
                INC     DPTR
                DJNZ    R1,agre_s126

                MOV     DPTR,#men_cargaok
                LCALL   s_lcd_romr1
                RET
```

```
;&&     ==============================
;       s_ehm
;       eliminar, definir hot, definir abreviado
;       teclado I
;
;       funciones:
;       1.-     seleccionar el abonado
;       2.-     [*] salir a modo programacion
;       3.-     [#] ejecuta funcion escogida en p_prog: eliminar, definir
;               abonado hot, definir abonado abreviado
;       ==============================

s_ehm:
ehm_s0:         CLR     f_tpos                  ; setear contador y bandera
                MOV     R3,#00H                 ; de las teclas

                CLR     f_dp                    ; mostrar el primer abonado
                MOV     dir_pun,#01H            ; del directorio
                AJMP    ehm_s4

ehm_s2:         LCALL   s_find
                JNB     f_nf,ehm_s4             ; f_nf=1 => no encontrol
                LCALL   s_lcd_nf                ; grupo sin abonados
                AJMP    ehm_s6

ehm_s4:         LCALL   s_lcd_ram
ehm_s6:         LCALL   s_tecla0                ; teclado I: elim/hotl/mabr
                JB      ix_tel1,ehm_s10
                RET

ehm_s10:        CJNE    A,#2AH,ehm_s20          ; [*] salir
                RET

ehm_s20:        CJNE    A,#23H,ehm_s30          ; [#] ejecutar funcion
                MOV     A,numero
                JZ      ehm_s6                  ; nada que ejecutar

                JNB     f_elim,ehm_s22          ; direccionar funcion
                AJMP    s_elim
ehm_s22:        JNB     f_hotl,ehm_s24
                LJMP    s_hotl
ehm_s24:        LJMP    s_mabr

ehm_s30:        LCALL   s_t_alfn                ; teclado alfab y de naveg
                JNB     f_alf,ehm_s4            ; f_alf=1 => alfabetico
                AJMP    ehm_s2                  ; f_alf=0 => navegacion
```

```
;&&     ------------------------------
;       s_elim
;       eliminar un abonado
;
;       elimina del directorio el bloque (nombre y numero) del abonado
;       seleccionado en s_ehm y definido por f_dp + dir-pun. La eliminacion
;       se ejecuta desplazando un bloque hacia atras todos los abonados del
;       directorio sobreescribiendo con el siguiente, el eliminado.
;       ------------------------------

s_elim:
elim_s0:        LCALL   s_add_ab                ; bloque destino
                MOV     stk_dph,DPH
                MOV     stk_dpl,DPL

                LCALL   s_inc_dp                ; sig bloque sera origen
                LCALL   s_add_ab

elim_s2:        MOVX    A,@DPTR
                INC     DPTR
                PUSH    DPH
                PUSH    DPL

                MOV     DPH,stk_dph             ; pasar el dato el bloque
                MOV     DPL,stk_dpl             ; origen (siguiente) al
                MOVX    @DPTR,A                 ; destino (actual)
                CJNE    A,#car_findir,elim_s6
                MOV     R0,#by_blo-1            ; fin de dir hallado
elim_s4:        INC     DPTR                    ; limpiar el ultimo bloque
                MOVX    @DPTR,A
                DJNZ    R0,elim_s4
                CLR     f_elim

elim_s6:        INC     DPTR                    ; el byte de bloque vacio
                MOV     stk_dph,DPH
                MOV     stk_dpl,DPL

                POP     DPL
                POP     DPH
                JB      f_elim,elim_s2

                MOV     DPTR,#men_elimok
                LCALL   s_lcd_romr1
                RET
```

```
;&&     ------------------------------
;       s_hotl
;       definir abonado hot
;       teclado N
;
;       Valor del ndc segun la linea conmutada directa
;               ndc: lsb-lsn    tipo de abonado
;               08H             hl1
;               0CH             hl2
;
;       funciones
;       1.- Teclado N:
;               [1]     definicion sobre lt1
;               [2]     definicion sobre lt2
;               [#]     indicar durante 1 segundo que lineas se hallan
;                       definidas como HOT (solo antes de [1] .OR. [2])
;       2.- Definicion y cambio de un abonado hot: si el abonado escogido con
;           s_ehm tiene en su ndc un valor diferente al indicado arriba:
;               - busca en el directorio si hay un abonado cuyo ndc coincida
;                 con los valores mencionados; de haberlo, lo define como
;                 normal cargando el valor 0AH en el ndc
;               - carga en el ndc del abonado escogido con s_ehm el valor
;                 indicado para definir a la linea como conmutada directa con
;                 el como abonado HOT.
;       3.- Eliminacion: si el abonado escogido con s_ehm tiene en su ndc un
;           valor igual al indicado en la tabla, lo que implica que este
;           abonado fue seleccionado previamente como HOT, se carga en su ndc
;           el valor 0AH (el abonado se define como normal) para indicar que
;           la linea ya NO es conmutada directa.
;       ------------------------------

s_hotl:
hotl_s0:        MOV     A,#0EH                  ; cursor on, blink off
                LCALL   s_lcd_wrd

hotl_s2:        MOV     A,#0CDH                 ; limpiar numero de abonado
                MOV     DPTR,#men_clrhotl
                LCALL   s_lcd_romx

                MOV     A,#0CFH                 ; cursor al final del display
                LCALL   s_lcd_wrd
                LCALL   s_tecla5                ; teclado N: definir lin hot
                JB      ix_tel1,hotl_s4
                RET

hotl_s4:        CJNE    A,#3AH,hotl_s10         ; [#] indicar lineas hot
                LCALL   s_lcd_hld               ; definidas
                LCALL   s_lcd_ram
                AJMP    hotl_s2

hotl_s10:       CJNE    A,#31H,hotl_s12         ; [1] hot para lt1
                SETB    f_cndc                  ; f_cndc=1 => def abonado normal
                LCALL   s_fhl1
                AJMP    hotl_s20

hotl_s12:       CJNE    A,#32H,hotl_s40         ; [2] hot para lt2
                SETB    f_cndc                  ; f_cndc=1 => def abonado normal
                LCALL   s_fhl2

;       ********        comparar
;       en f_dpbk + dir_punbk : abonado seleccionado con s_ehm
;       en f_dp + dir_pun : abonado existente en el directorio
;       esto son datos devueltos por s_fhl1 .OR. s_fhl2

hotl_s20:       JNB     f_fndc,hotl_s32         ; f_fndc=0 => ndc no encontrado
                MOV     A,dir_punbk             ; existe ab hot en el dir
                CJNE    A,dir_pun,hotl_s32      ; comparar hallado con seleccionado
                MOV     C,f_dpbk
                JB      f_dp,hotl_s22
                JNC     hotl_s30
                AJMP    hotl_s32

hotl_s22:       JNC     hotl_s32
```

```
;       ********        seleccionado el mismo: deshabilitar hot
hotl_s30:       MOV     DPTR,#men_hotoff
                LCALL   s_lcd_romr1
                RET

;       ********        seleccionado diferente: habilitar hot
hotl_s32:       LCALL   s_rndc
                MOV     DPTR,#men_hoton
                LCALL   s_lcd_romr1
                RET

hotl_s40:       LCALL   s_beep                  ; tecla sin funcion
                AJMP    hotl_s2
```

```
;&&     ----------------------------
;       s_mabr
;       definir abreviado
;       teclado J
;
;       Valor del ndc segun el numero del abonado abreviado:
;               ndc: lsb-lsn    tipo de abonado
;               0H              abr 0
;               1H              abr 1
;               2H              abr 2
;               3H              abr 3
;               4H              abr 4
;               5H              abr 5
;               6H              abr 6
;               7H              abr 7
;               8H              abr 8
;               9H              abr 9
;
;       funciones:
;       1.- [#] Indica por un segundo los numeros abreviados ya asignados
;               (solo antes de [0]...[9])
;       2.- [0]...[9] Esperar el numero del abonado abreviado decidido por
;                     el usuario.
;       3.- Definicion (o cambio) de un abonado abreviado.
;             - Busca en el directorio si ya hay un abonado definido con el
;               numero escogido; de haberlo, lo defino como normal escribiendo
;               en su ndc el valor 0AH.
;             - En el ndc del abonado escogido con s_ehm se carga el numero
;               ingresado por el usuario.
;
;       Notar:  La forma de eliminar un abonado abreviado es borrarlo del
;               directorio o asignar su numero a otro abonado.
;       ----------------------------

s_mabr:
mabr_s0:        MOV     A,#0EH                  ; cursor on, blink off
                LCALL   s_lcd_wrd

mabr_s2:        MOV     A,#0CDH                 ; limpiar numero de abonado
                MOV     DPTR,#men_clrmabr
                LCALL   s_lcd_romx

mabr_s4:        MOV     A,#0CFH                 ; cursor al final del display
                LCALL   s_lcd_wrd
                LCALL   s_tecla5                ; teclado J: definir # del abr
                JB      ix_tel1,mabr_s6
                RET

mabr_s6:        CJNE    A,#20H,mabr_s8          ; [*] sin funcion
                LCALL   s_beep
                AJMP    mabr_s4

mabr_s8:        CJNE    A,#3AH,mabr_s10         ; [#] indicar numeros abreviados
                LCALL   s_lcd_abrap             ; ya asignados
                LCALL   s_lcd_ram
                AJMP    mabr_s2

mabr_s10:       ANL     A,#0FH
                SETB    f_cndc                  ; si hay un abonado definido como abr
                LCALL   s_fabr                  ; con ese numero, se lo cambia a normal
                LCALL   s_rndc
                RET
```

```
;&&     ==============================
;       s_line
;       linea - equipo
;       teclado: K
;
;       funciones:
;       -       permitir la definicion de:
;               tipo de linea:          tonos - pulsos,
;               bloqueos:               lt1:            ldi, ldn
;                                       lt2:            total, ldi, ldn
;               equipo:                 normalmente encendido o apagado
;       Presionando la tecla correspondiente se habilita o deshabilita la
;       funcion indicada:
;               lt1:            [1] pulsos - tonos
;                               [2] bloqueo ldi
;                               [3] bloqueo ldn
;               lt2:            [4] pulsos - tonos
;                               [5] bloqueo ldi
;                               [6] bloqueo ldn
;                               [7] bloqueo total
;               EQUIPO:         [8] normalmente encendido - normalmente apagado
;       (los cambios hechos en bloqueos se hacen efectivos luego de que el
;       equipo vuelva al estado de espera y de que lt2 se halle desocupada,
;       si esta en uso en el momento de programar)
;
;       [0]     reset la configuracion: lineas de tonos, ningun bloqueo,
;               equipo normalmente apagado
;       [*]     cargar cambios en memoria
;       ==============================

s_line:         MOV     DPTR,#men_cnf
                LCALL   s_lcd_rom
line_s2:        LCALL   s_lcd_cnfl              ; mostrar configuracion

line_s10:       LCALL   s_tecla0                ; teclado K: linea-equipo
                JB      ix_tel1,line_s12
                RET

line_s12:       CJNE    A,#20H,line_s14         ; [1] f_pt1
                CPL     f_pt1
                AJMP    line_s2

line_s14:       CJNE    A,#41H,line_s16         ; [2] f_bi1
                CPL     f_bi1
                AJMP    line_s2

line_s16:       CJNE    A,#44H,line_s18         ; [3] f_bn1
                CPL     f_bn1
                AJMP    line_s2

line_s18:       CJNE    A,#47H,line_s20         ; [4] f_pt2
                CPL     f_pt2
                AJMP    line_s2

line_s20:       CJNE    A,#4AH,line_s22         ; [5] f_bi2
                CPL     f_bi2
                AJMP    line_s2

line_s22:       CJNE    A,#4DH,line_s24         ; [6] f_bn2
                CPL     f_bn2
                AJMP    line_s2

line_s24:       CJNE    A,#50H,line_s26         ; [7] f_bt2
                CPL     f_bt2
                AJMP    line_s2

line_s26:       CJNE    A,#54H,line_s28         ; [8] f_ne
                CPL     f_ne
                AJMP    line_s2

line_s28:       CJNE    A,#30H,line_s30         ; [0] clear all
                MOV     flagic0,#00H
                AJMP    line_s2
```

```
line_s30:       CJNE    A,#2AH,line_s40         ; [*] salvar cambios
                MOV     A,flagic0
                MOV     DPTR,#flagxc0
                MOVX    @DPTR,A

                JB      f_bt2,line_s32
                LCALL   c_r_lt2                 ; deshabil bloqueo total en lt2
                LCALL   s_r_flt2                ; deshabil filtro lt2
                AJMP    line_s34

line_s32:       LCALL   c_r_flt2                ; habil filtro lt2
                LCALL   s_r_lt2                 ; habil bloqueo total en lt2

line_s34:       MOV     DPTR,#men_cargaok
                LCALL   s_lcd_romr1
                RET

line_s40:       LCALL   s_beep                  ; tecla sin funcion
                AJMP    line_s10
```

```
;&&     ============================
;       s_tiem
;       tiempo (fecha - hora) del equipo
;
;       funcion:
;       Permite el ingreso de una nueva fecha y hora para el equipo.
;       -       El tiempo del sistema consta de 10 digitos que deberan ser
;               ingresados en el siguiente orden y en los rangos indicados:
;                       orden                   rango
;                       1 mes                   00 - 12
;                       2 dia del mes           01 - 31
;                       3 dia de la semana      01 - 07
;                       4 hora                  00 - 23
;                       5 minuto                00 - 59
;               La numeracion para el dia de la semana es como sigue:
;                       lunes           01
;                       martes          02
;                       miercoles       03
;                       jueves          04
;                       viernes         05
;                       sabado          06
;                       domingo         07
;       -       al inicio del la carga, o al volver al inicio, se indica el
;               tiempo del sistema
;       -       durante el ingreso de datos:
;               [*]     backspace, vuelve al inicio del ingreso
;               [#]     sin funcion
;       -       una vez ingresados todos los datos:
;               [*]     backspace, vuelve al inicio del ingreso
;               [#]     cargar el nuevo tiempo; antes de la carga se comprueban
;                       los rango indicados (no la fecha, p.e. 31 de febrero);
;                       si son correctos se actualiza el tiempo, si no, se da
;                       un mensaje de error indicando su tipo y regresando al
;                       inicio del ingreso.
;       La unica forma de salir sin cargar una nueva fecha es colgando el
;       telefono.
;       ============================

s_tiem:         MOV      DPTR,#men_tiemin
                LCALL   s_lcd_rom
                MOV     A,#0EH                  ; cursor on, blink off
                LCALL   s_lcd_wrd

;       ********        mostrar tiempo del equipo
tiem_s2:        MOV     A,#0C0H                 ; cursor al inicio de la
                LCALL   s_lcd_wrd               ; segunda fila

                MOV     DPTR,#c_mes             ; sacar fecha al display
tiem_s4:        LCALL   s_lcd_bcdd
                DEC     DPL
                MOV     A,DPL
                CJNE    A,#c_seg,tiem_s4
                LCALL   s_ret_1000

;       ********        ingresar tiempo
tiem_s10:       MOV     R0,#numero
                MOV     A,#0C0H                 ; cur al ini de la seg fila
                LCALL   s_lcd_wrd

tiem_s12:       CJNE    R0,#numero+0AH,tiem_s14
                AJMP    tiem_s40                ; 10 numeros a ingresar

tiem_s14:       MOV     A,#20H                  ; espacio en blanco
                LCALL   s_lcd_wr
                MOV     A,#80H                  ; regresar cursor
                ADD     A,R0
                LCALL   s_lcd_wrd

                LCALL   s_tecla5                ; teclado L: tiempo
                JB      ix_tel1,tiem_s16
                RET
```

```
tiem_s16:       CJNE    A,#3AH,tiem_s20         ; [#] sin funcion
                MOV     A,#80H
                ADD     A,R0
                LCALL   s_lcd_wrd
                LCALL   s_beep
                AJMP    tiem_s14

tiem_s20:       CJNE    A,#20H,tiem_s30         ; [*] backspace
                AJMP    tiem_s2

tiem_s30:       MOV     @R0,A                   ; [0] - [9]: cargar
                INC     R0
                AJMP    tiem_s12

;       ********        ingreso concluido
tiem_s40:       MOV     DPTR,#men_ok
                LCALL   s_lcd_romy

tiem_s42:       LCALL   s_tecla0                ; teclado M: tiempo concluido
                JB      ix_tel1,tiem_s44
                RET

tiem_s44:       CJNE    A,#2AH,tiem_s50         ; [*] backspace
tiem_s46:       MOV     A,#0CAH
                MOV     DPTR,#men_clrrel
                LCALL   s_lcd_romx
                AJMP    tiem_s2

;       ********        cargar
tiem_s50:       CJNE    A,#23H,tiem_s42         ; [#] cargar
                MOV     R0,#numero-01H
                MOV     R1,#numero+09H

tiem_s52:       INC     R0                      ; paso a BCD
                INC     R1
                MOV     A,@R0
                ANL     A,#0FH
                SWAP    A
                INC     R0
                MOV     B,@R0
                ANL     B,#0FH
                ORL     A,B
                MOV     @R1,A
                CJNE    R0,#numero+09H,tiem_s52
                INC     R0

;       ********        comprobar rangos
                CJNE    @R0,#13H,tiem_s60       ; 00 < mes < 13
tiem_s60:       JNC     tiem_s80
                MOV     A,@R0
                JZ      tiem_s80
                INC     R0

                CJNE    @R0,#32H,tiem_s62       ; 00 < dim < 32
tiem_s62:       JNC     tiem_s80
                MOV     A,@R0
                JZ      tiem_s80
                INC     R0

                CJNE    @R0,#08H,tiem_s64       ; 00 < dis < 08
tiem_s64:       JNC     tiem_s80
                MOV     A,@R0
                JZ      tiem_s80
                INC     R0

                CJNE    @R0,#24H,tiem_s66       ; hh < 24
tiem_s66:       JNC     tiem_s80
                INC     R0

                CJNE    @R0,#60H,tiem_s68       ; mm < 60
tiem_s68:       JNC     tiem_s80
```

```
;       ********        datos correctos
                INC     R0                      ; datos correctos
                MOV     @R0,#00H                ; clr seg
                MOV     DPTR,#c_seg
tiem_s70:       MOV     A,@R0
                MOVX    @DPTR,A
                INC     DPTR
                DEC     R0
                CJNE    R0,#numero+09H,tiem_s70

                LCALL   s_feho
                LCALL   s_ret_max
                MOV     DPTR,#men_cargaok
                LCALL   s_lcd_romr1
                RET

;       ********        datos incorrectos
tiem_s80:       MOV     A,#0CAH                 ; datos incorrectos
                MOV     DPTR,#men_er
                LCALL   s_lcd_romx

                MOV     A,R0                    ; indicar tipo de error
                ADD     A,#0B6H
                RL      A
                PUSH    A
                MOV     DPTR,#men_tiemin
                MOVC    A,@A+DPTR
                LCALL   s_lcd_wr
                POP     A
                INC     A
                MOVC    A,@A+DPTR
                LCALL   s_lcd_wr
                LCALL   s_alarma0
                AJMP    tiem_s46
```

```
;&&     ************************************************************
;                       ALARMA
;       ************************************************************

;       ==============================
;       s_alarma
;       s_alarmax
;       s_alarmay
;
;       saca al display el mensaje apuntado por DPTR mediante las subrutinas
;       s_lcd_rom, s_lcd_romx o s_lcd_romy respectivamente, junto con un grupo
;       de beeps
;       ==============================
s_alarmax:      LCALL   s_lcd_romx
                AJMP    s_alarma0

s_alarmay:      LCALL   s_lcd_romy
                AJMP    s_alarma0

s_alarma:       LCALL   s_lcd_rom
s_alarma0:      LCALL   s_beep
                LCALL   s_beep
                LCALL   s_beep
                LCALL   s_ret_1000
                RET



;       ==============================
;       s_alarmas
;       alarma para el modo serial
;
;       saca al display el mensaje apuntado por DPTR mediante la subrutina
;       s_lcd_rom junto con un beep
;       ==============================
s_alarmas:      LCALL   s_lcd_rom
                LCALL   s_beep
                LCALL   s_ret_500
                RET



;       ==============================
;       s_beep
;       beeper
;
;       emite un tono de cadencia:
;               tono    100ms
;               pausa    50ms
;       ==============================
s_beep:         CLR     ix_beep
                LCALL   s_ret_100
                SETB    ix_beep
                LCALL   s_ret_50
                RET
```

```
;&&     ************************************************************
;                       DIRECTORIO
;       ************************************************************

;       ==============================
;       s_add_ab
;       direccion de un abonado
;
;       Posiciona al DPTR en la primera localidad del bloque del abonado
;       definido por f_dp + dir_pun.
;       La direccion es calculada multiplicando el valor del
;       puntero (f_dp + dir_pun) por el numero de bytes por bloque.
;
;       IN      dir_pun, f_dp
;       OUT     DPTR
;       ==============================
s_add_ab:       MOV     B,#by_blo
                MOV     A,dir_pun
                MUL     AB
                MOV     DPL,A
                JB      f_dp,add_ab_s2
                MOV     DPH,B
                RET

add_ab_s2:      MOV     A,B
                ADD     A,#by_blo
                MOV     DPH,A
                RET



;&&     ==============================
;       s_clr_dir
;       limpiar directorio
;
;       Borrado manual o automatico de la memoria de datos.
;       - Automatico: ssi memoria no seteada
;                     definir: lineas de tonos, no bloqueos, equipo NA,
;                     borrar directorio
;       - Manual:     ssi llave puesta (informacion desprotegida)
;                     borrar directorio
;
;       FLAG  f_clraut
;       ==============================
s_clr_dir:      CLR     f_clraut
                MOV     DPTR,#mset0             ; revisa si la memoria ya ha
                MOVX    A,@DPTR                 ; sido seteada
                CJNE    A,#car_mset,clr_dir_s4
                INC     DPTR
                MOVX    A,@DPTR
                CJNE    A,#car_mset,clr_dir_s4
                JB      ix_sw1,clr_dir_s6       ; pedido de limpieza manual?
clr_dir_s2:     RET

clr_dir_s4:     SETB    f_clraut                ; ordenar limpieza automatica
clr_dir_s6:     MOV     DPTR,#men_clrdir

                JNB     f_clraut,clr_dir_s8
                LCALL   s_lcd_rom
                MOV     DPTR,#men_autom         ; ejecutar limpieza automatica
                LCALL   s_alarmay
                MOV     DPTR,#flagxc0           ; lineas de tonos,
                CLR     A                       ; eliminar bloqueos y eq NA
                MOVX    @DPTR,A
                AJMP    clr_dir_s20

clr_dir_s8:     JNB     ix_lv,clr_dir_s30       ; llave puesta?
                LCALL   s_alarma
                MOV     A,#0DBH
                MOV     R0,#10H
clr_dir_s10:    LCALL   s_lcd_wr
                LCALL   s_ret_250               ; sw1 presionado por 4 seg
                JNB     ix_sw1,clr_dir_s2       ; para ejecutar limpieza manual
```

```
                DJNZ    R0,clr_dir_s10

clr_dir_s20:    MOV     DPTR,#mset0             ; borrar car de mem set y nombre
                MOV     A,#20H                  ; del directorio de trabajo
                MOV     R0,#0AH
clr_dir_s22:    MOVX    @DPTR,A
                INC     DPTR
                DJNZ    R0,clr_dir_s22

                MOV     A,#car_findir           ; limpiar contenido del directorio
clr_dir_s24:    MOVX    @DPTR,A
                INC     DPTR
                MOV     R4,DPH
                CJNE    R4,#20H,clr_dir_s24     ; para 8Kx8

                MOV     DPTR,#mset0             ; poner indicativo de memoria seteada
                MOV     A,#car_mset
                MOVX    @DPTR,A
                INC     DPTR
                MOVX    @DPTR,A

                MOV     DPTR,#men_clrfin
                LCALL   s_lcd_romr1
                RET

clr_dir_s30:    LCALL   s_lcd_rom
                MOV     DPTR,#men_infpc         ; informacion protegida,
                LCALL   s_alarmay               ; no borrar el directorio

clr_dir_s32:    NOP                             ; sale cuando sw1 libre
                JB      ix_sw1,clr_dir_s32
                RET



;&      ==============================
;       s_find
;       find
;
;       Busca el primer abonado del grupo definido por el caracter hallado en
;       A; si hay, indica en el puntero (f_dp + dir_pun) el bloque hallado y
;       con DPTR la direccion del primer caracter del bloque; si no, setea la
;       bandera f_nf y direcciona el primer caracter del bloque del abonado
;       siguiente; carga el grupo buscado en la localidad "grupo".
;
;       IN      A
;       OUT     dir_pun, f_dp, DPTR, f_nf, grupo
;       ==============================
s_find:         MOV     grupo,A
                MOV     dir_pun,#00H            ; puntero al inicio del dir
                CLR     f_dp
                CLR     f_nf

find_s2:        LCALL   s_inc_dp                ; incrementar puntero
                LCALL   s_add_ab
                MOVX    A,@DPTR
                CJNE    A,grupo,find_s4
                RET

find_s4:        JC      find_s2
                SETB    f_nf                    ; si (A)>(grupo) => C = 0
                RET                             ; no hay abonados en el grupo
```

```
;&&     ==============================
;       s_fhl1 / s_fhl2 / s_fabr
;       s_fndc
;       find ...
;
;       s_fhl1: busca el abonado hot para la linea 1.
;       s_fhl2: busca el abonado hot para la linea 2.
;       s_fabr: busca el abonado abreviado indicado con A.
;       s_fndc: busca el abonado cuyo nibble de configuracion coincide con el
;               valor de la localidad val_ndc; utilizando s_add_ab para el
;               abonado n+1 y decrementando en 1 el DPTR llega al byte que
;               contiene el nibble de configuracion del abonado n. Devuelve en
;               f_dpbk + dir_punbk la direccion del abonado apuntador al
;               momento de ejecutar la subrutina, y, en val_ndc el valor de
;               ndc buscado.
;               Si encuentra:
;                       f_fndc = 1
;                       dir_pun y f_dp abuntan al abonado
;                       DPTR apunta al byte que contiene el ndc
;                       define el abonado como normal si f_cndc = 1
;               Si no encuentra:
;                       f_fndc = 0
;
;       FLAG    f_fndc
;       ==============================
s_fhl1:         MOV     val_ndc,#ndchl1
                AJMP    s_fndc

s_fhl2:         MOV     val_ndc,#ndchl2
                AJMP    s_fndc

s_fabr:         MOV     val_ndc,A

s_fndc:         MOV     dir_punbk,dir_pun       ; salvar dir del ab actual
                MOV     C,f_dp
                MOV     f_dpbk,C

                MOV     dir_pun,#01H            ; puntero al BOF
                CLR     f_dp

fndc_s2:        LCALL   s_sfndc
                ANL     A,#0FH

                CJNE    A,#0EH,fndc_s4          ; fin del directorio?
                CLR     f_fndc                  ; no encontro ndc igual
                RET

fndc_s4:        CJNE    A,val_ndc,fndc_s2
                LCALL   s_dec_dp                ; decrementar puntero del dir
                SETB    f_fndc                  ; encontrado ndc igual

                JNB     f_cndc,fndc_s6
                MOVX    A,@DPTR                 ; definir como normal al abonado
                ANL     A,#0F0H
                ORL     A,#ndcnor
                MOVX    @DPTR,A
fndc_s6:        RET
```

```
;&&     ==============================
;       s_sfndc
;       subrutina de la find nibble de configuracion
;
;       Devuelve en DPTR la direccion del byte que contiene el ncd del abonado
;       apuntado, en A el valor de dicho byte e incrementa en 1 el puntero del
;       directorio.
;
;       IN      dir_pun, f_dp
;       ==============================
s_sfndc:        LCALL   s_inc_dp                ; incrementar puntero
                LCALL   s_add_ab
                LCALL   s_dec_dptr              ; dec DPTR
                MOVX    A,@DPTR
                RET



;&&     ==============================
;       s_dec_dp
;       decrementar puntero del directorio
;
;       Decrementa en uno el puntero del directorio (f_dp + dir_pun).
;       ==============================
s_dec_dp:       MOV     A,dir_pun
                JNZ     dec_dp_s2
                CPL     f_dp
dec_dp_s2:      DEC     dir_pun
                RET



;       ==============================
;       s_inc_dp
;       incrementar puntero del directorio
;
;       Incrementa en uno el puntero del directorio (f_dp + dir_pun).
;       ==============================
s_inc_dp:       INC     dir_pun
                MOV     A,dir_pun
                JNZ     inc_dp_s2
                CPL     f_dp
inc_dp_s2:      RET



;       ==============================
;       s_dec_dptr
;       decrementar puntero del directorio
;
;       Decrementa en uno el puntero de datos (DPTR).
;       ==============================
s_dec_dptr:     MOV     A,DPL
                JNZ     dec_dptr_s2
                DEC     DPH
dec_dptr_s2:    DEC     DPL
                RET
```

```
;       ==============================
;       s_rndc
;       reemplazar nibble de configuracion
;
;       Guarda en el ndc del abonado apuntado por dir_punbk y f_dpbk el valor
;       hallado en val_ndc.
;       ==============================
s_rndc:         MOV     C,f_dpbk                ; localizar ndc del
                MOV     f_dp,C                  ; ab seleccionado
                MOV     dir_pun,dir_punbk
                LCALL   s_sfndc
                ANL     A,#0F0H                 ; cargar en el ndc del ab
                ORL     A,val_ndc               ; seleccionado el valor
                MOVX    @DPTR,A                 ; hallado en val_ndc
                RET
```

```
;&&     ************************************************************
;                       display
;       ************************************************************

;       ==============================
;       p_lcd_ini
;       display_inicializacion
;
;       Inicializacion del display con:
;               interfaz 4 bits, dual line, 5x7,
;               display on, cursor off, blink off,
;               incrementar, not shift
;       ==============================
p_lcd_ini:      LCALL   s_beep                  ; retardo = 150ms
                MOV     R0,#0FFH                ; R0 puntero/contador
                MOV     DPTR,#tab_ini

lcd_ini_s2:     INC     R0
                MOV     A,R0
                MOVC    A,@A+DPTR
                LCALL   s_lcd_wrd
                CJNE    R0,#05H,lcd_ini_s2

                MOV     DPTR,#men_ini           ; mensaje inicial
                LCALL   s_lcd_romrm

                MOV     DPTR,#men_nomeq         ; nombre del equipo
                LCALL   s_lcd_romrm

                MOV     DPTR,#men_ver           ; version
                LCALL   s_lcd_romrm
                RET



;&&     ==============================
;       s_lcd_abrap
;       s_lcd_abra
;       display abreviados asignados (modo programacion)
;
;       Indica los abonados abreviados que hay en el directorio
;               si encuentra abreviados:        f_eaua = 1
;               si NO encuentra:                f_eaua = 0
;       No altera el valor del ndc ni del puntero del directorio
;
;       FLAG    f_eaua, f_lcdabrp
;       ==============================
s_lcd_abrap:    SETB    f_lcdabrp               ; modo programacion
                LCALL   s_lcd_clr
                MOV     A,#84H
                MOV     DPTR,#men_abr+07H
                LCALL   s_lcd_romx
                MOV     A,#0C3H                 ; cursor a la segunda fila
                LCALL   s_lcd_wrd
                AJMP    lcd_abra_s0

s_lcd_abra:     CLR     f_lcdabrp               ; modo equipo activo, marcar
                MOV     DPTR,#men_abr           ; abreviado
                LCALL   s_lcd_rom

lcd_abra_s0:    CLR     f_cndc
                CLR     f_eaua
                MOV     R0,#val_ndc
                MOV     @R0,#0FFH

lcd_abra_s2:    INC     @R0                     ; INC val_ncd
                LCALL   s_fndc
                MOV     dir_pun,dir_punbk       ; no alterar puntero del dir
                MOV     C,f_dpbk
                MOV     f_dp,C

                JB      f_fndc,lcd_abra_s4
                MOV     A,#0A5H                 ; si NO encuentra escribir "."
```

```
                AJMP    lcd_abra_s6

lcd_abra_s4:    SETB    f_eaua
                MOV     A,val_ndc
                ORL     A,#30H                  ; si encuentra escribe ndc
lcd_abra_s6:    LCALL   s_lcd_wr
                CJNE    @R0,#09H,lcd_abra_s2

                JB      f_lcdabrp,lcd_abra_s10
                MOV     A,#0C2H                 ; cursor para escribir
                LCALL   s_lcd_wrd               ; numero seleccionado
                RET

lcd_abra_s10:   LCALL   s_ret_max               ; consulta en modo prog
                RET



;&&     ==============================
;       s_lcd_bcda / s_lcd_bcdd
;       display_bcd
;
;       Escribe en el display los dos numeros contenidos en un byte con
;       formato BCD.
;       s_lcd_bcda: el numero se halla en el acumulador A
;       s_lcd_bcdd: el numero se halla en RAMx, apuntado por DPTR
;
;       IN      A, DPTR
;       ==============================
s_lcd_bcdd:     MOVX    A,@DPTR
s_lcd_bcda:     SWAP    A
lcd_bcd_s2:     ANL     A,#0FH                  ; pasar a ASCII
                ORL     A,#30H
                LCALL   s_lcd_wr                ; sacar a display

                JBC     f_l4,lcd_bcd_s4
                SETB    f_l4                    ; lsn cargado
                MOVX    A,@DPTR
                AJMP    lcd_bcd_s2

lcd_bcd_s4:     RET



;&&     ==============================
;       s_lcd_ck1
;       display_clock1
;
;       Escribe en el display los tres primeros caracteres del nombre del mes
;       o dia de la semana, segun la tabla apuntada por DPTR (mes o dia) y el
;       contenido de A (cual mes o cual dia)
;
;       IN      A, DPTR
;       ==============================
s_lcd_ck1:      DEC     A
                RL      A
                RL      A
                MOV     R2,A
                MOV     R1,#04H         ; presentar 4 caracteres de la tabla

lcd_ck1_s2:     MOV     A,R2
                MOVC    A,@A+DPTR
                LCALL   s_lcd_wr
                INC     R2
                DJNZ    R1,lcd_ck1_s2
                RET
```

```
;       ==============================
;       s_lcd_ck2
;       display_clock2
;
;       Escribe en el display el numero del dia, hora, minuto o segundo
;       hallado en @R0, pasando previamente de BCD a ASCII
;
;       IN      A
;       FLAG    f_l4
;       ==============================
s_lcd_ck2:      MOV     A,@R0
                SWAP    A
lcd_ck2_s2:     ANL     A,#0FH
                ORL     A,#30H                  ; pasar a ASCII
                LCALL   s_lcd_wr

                JBC     f_l4,lcd_ck2_s4
                SETB    f_l4
                MOV     A,@R0
                AJMP    lcd_ck2_s2

lcd_ck2_s4:     RET



;&&     ==============================
;       s_lcd_cln
;       display clear numero
;
;       borrar la fila 2 y dejar el cursor al inicio para indicar numero
;       marcado
;       ==============================
s_lcd_cln:      MOV     A,#0C0H                 ; cursor al inicio 2da fila
                MOV     DPTR,#men_clrfil        ; borrar fila 2
                LCALL   s_lcd_romx

                MOV     A,#0C0H                 ; cursor al inicio 2da fila
                LCALL   s_lcd_wrd
                MOV     A,#7EH                  ; flecha
                LCALL   s_lcd_wr
                RET



;       ==============================
;       s_lcd_clr
;       display_clear
;
;       Borra el display y posiciona el cursor en la localidad 00H
;       ==============================
s_lcd_clr:      MOV     A,#01H
                LCALL   s_lcd_wrd
                LCALL   s_ret_10                ; retardo = 10ms
                RET
```

```
;&&     ==============================
;       s_lcd_cnf / s_lcd_cnfl
;       display_configuracion
;
;       Muestra la configuracion del equipo: bloqueos sobre las lineas y
;       estado del equipo (normalmente encendido o apagado)
;
;       s_lcd_cnfl no muestra el mensaje en la primera linea, solo la
;               configuracion.
;
;       IN      flagic0
;       ==============================
s_lcd_cnf:      MOV     DPTR,#men_cnf
                LCALL   s_lcd_rom

s_lcd_cnfl:     MOV     DPTR,#tab_cnf
                MOV     A,flagic0
                MOV     R0,#00H
lcd_cnf_s2:     RRC     A
                PUSH    A

                MOV     A,R0
                MOVC    A,@A+DPTR
                LCALL   s_lcd_wrd
                JC      lcd_cnf_s4      ; si la opcion esta activa se
                MOV     A,#20H          ; muestra un '*'
                AJMP    lcd_cnf_s6

lcd_cnf_s4:     MOV     A,#2AH
lcd_cnf_s6:     LCALL   s_lcd_wr

                INC     R0
                POP     A
                CJNE    R0,#09H,lcd_cnf_s2
                RET



;&&     ==============================
;       s_lcd_hld
;       display lineas hot definidas
;
;       Indica cuales lineas se hallan definidas como conmutadas directas.
;       No altera el contenido del puntero del directorio.
;       ==============================
s_lcd_hld:      LCALL   s_lcd_clr
                MOV     DPTR,#men_hld
                LCALL   s_lcd_rom

                CLR     f_cndc
                LCALL   s_fhl1                  ; lt1
                MOV     dir_pun,dir_punbk       ; no alterar puntero del dir
                MOV     C,f_dpbk
                MOV     f_dp,C

                JNB     f_fndc,lcd_hld_s2
                MOV     DPTR,#men_lt1           ; lt1 es hot
                AJMP    lcd_hld_s4

lcd_hld_s2:     MOV     DPTR,#men_hlnd          ; lt1 NO es hot
lcd_hld_s4:     LCALL   s_lcd_romy

                LCALL   s_fhl2                  ; lt2
                MOV     dir_pun,dir_punbk       ; no alterar puntero del dir
                MOV     C,f_dpbk
                MOV     f_dp,C

                JNB     f_fndc,lcd_hld_s6
                MOV     DPTR,#men_lt2           ; lt2 es HOT
                AJMP    lcd_hld_s8

lcd_hld_s6:     MOV     DPTR,#men_hlnd          ; lt2 NO es hot
lcd_hld_s8:     LCALL   s_lcd_romy
```

```
                LCALL   s_ret_max
                RET


;&&     ==============================
;       s_lcd_hms
;       display horas minutos segundos
;
;       Escribe en el display, a paratir de la localidad donde se halla el
;       cursor, la hora, el minuto y el segundo (en el formato 00h00m00s)
;       que se hallan en el banco formado por las localidades hhd, mmd y ssd.
;       ==============================
s_lcd_hms:      MOV     R0,#hhd                 ; hora
                LCALL   s_lcd_ck2
                MOV     A,#68H                  ; escribir 'h'
                LCALL   s_lcd_wr

                DEC     R0                      ; minuto
                LCALL   s_lcd_ck2
                MOV     A,#6DH                  ; escribir 'm'
                LCALL   s_lcd_wr

                DEC     R0                      ; segundos
                LCALL   s_lcd_ck2
                MOV     A,#73H                  ; escribir 's'
                LCALL   s_lcd_wr
                RET


;&&     ==============================
;       s_lcd_nf
;       display_escribir no hallado
;
;       Saca mensaje de GRUPO VACIO y posiciona el puntero al inicio de
;       directorio.
;
;       IN      grupo
;       OUT     numero
;       ==============================
s_lcd_nf:       MOV     A,#80H
                MOV     DPTR,#men_nf1           ; mensaje generico
                LCALL   s_lcd_romx

                MOV     A,grupo                 ; inicar grupo
                LCALL   s_lcd_wr

                MOV     DPTR,#men_nf2           ; mensaje generico
                LCALL   s_lcd_romy

                MOV     A,#0C0H
                MOV     DPTR,#men_nf3           ; mensaje generico
                LCALL   s_lcd_romx

                MOV     dir_pun,#01H            ; puntero al inicio del directorio
                CLR     f_dp
                MOV     numero,#00H             ; => ningun # en el banco
                LCALL   s_beep
                RET
```

```
;&&     ==============================
;       s_lcd_ram
;       display_escribir de ram
;
;       Muestra en el display el contenido del bloque apuntado (definido por
;       f_dp + dir_pun): nombre y numero del abonado, y el numero del bloque
;       (f_dp + dir_pun); carga en el banco del numero el ASCII correspondiente
;       al numero del abonado, listo para ser utilizado por s_mar.
;
;       IN      dir_pun, f_dp
;       OUT     banco del numero
;       FLAG    f_l4
;       ==============================
s_lcd_ram:      LCALL   s_add_ab

                MOV     A,#080H                 ; cursor a la inicio del display
                LCALL   s_lcd_wrd

                MOV     R2,#10H                 ; mostrar nombre (16 bytes)
lcd_ram_s2:     MOVX    A,@DPTR
                INC     DPTR
                LCALL   s_lcd_wr
                DJNZ    R2,lcd_ram_s2

                MOV     A,#0C0H                 ; cursor a la siguiente fila
                LCALL   s_lcd_wrd

                MOV     R1,#numero              ; puntero
                MOV     R2,#07H                 ; mostrar numero (7 bytes)

                MOVX    A,@DPTR                 ; si es ddi, carga 00 al inicio
                ANL     A,#0F0H
                CJNE    A,#0A0H,lcd_ram_s10
                MOV     @R1,#30H
                INC     R1
                MOV     @R1,#30H
                INC     R1
                MOV     A,#69H                  ; "i" en vez de "00" en el
                LCALL   s_lcd_wr                ; display
                AJMP    lcd_ram_s14

lcd_ram_s10:    MOVX    A,@DPTR
                SWAP    A
lcd_ram_s12:    ANL     A,#0FH
                ORL     A,#30H                  ; pasar a ASCII
                MOV     @R1,A                   ; cargar banco
                INC     R1
                LCALL   s_lcd_wr

                JBC     f_l4,lcd_ram_s16
lcd_ram_s14:    SETB    f_l4
                MOVX    A,@DPTR
                AJMP    lcd_ram_s12

lcd_ram_s16:    INC     DPTR
                DJNZ    R2,lcd_ram_s10

;       si el abonado es normal, indicar el valor decimal del puntero del
;       directorio, sino, indica que tipo de abonado especial es

                MOV     A,#0CDH                 ; numero de orden del abonado
                LCALL   s_lcd_wrd               ; pasar a decimal el f_dp dir_pun

                DEC     R1                      ; al final del banco esta el
                CJNE    @R1,#3AH,lcd_ram_s30    ; valor del ndc en ASCII

                MOV     C,f_dp
                MOV     A,dir_pun
                MOV     DPTR,#tab_hex_dec
                MOV     centena,#00H
                MOV     dec_uni,#00H
                MOV     R0,#09H                 ; contador
                MOV     R1,#0FFH                ; puntero de la tabla
```

```
lcd_ram_s20:    JNC     lcd_ram_s24
                CLR     C
                PUSH    A
                INC     R1
                MOV     A,R1
                MOVC    A,@A+DPTR
                ADD     A,dec_uni
                DA      A
                MOV     dec_uni,A
                INC     R1
                MOV     A,R1
                MOVC    A,@A+DPTR
                ADDC    A,centena
                DA      A
                MOV     centena,A
                POP     A
lcd_ram_s22:    RLC     A
                DJNZ    R0,lcd_ram_s20

                MOV     A,centena               ; escribir centena
                ORL     A,#30H
                LCALL   s_lcd_wr

                MOV     A,dec_uni               ; escribir decena
                SWAP    A
                ANL     A,#0FH
                ORL     A,#30H
                LCALL   s_lcd_wr

                MOV     A,dec_uni               ; escribir unidad
                ANL     A,#0FH
                ORL     A,#30H
                LCALL   s_lcd_wr
                RET

lcd_ram_s24:    INC     R1
                INC     R1
                AJMP    lcd_ram_s22

lcd_ram_s30:    CJNE    @R1,#3BH,lcd_ram_s32
                MOV     DPTR,#men_hl1d          ; abonado hot para lt1
                LCALL   s_lcd_romy
                RET

lcd_ram_s32:    CJNE    @R1,#3CH,lcd_ram_s34
                MOV     DPTR,#men_hl2d          ; abonado hot para lt2
                LCALL   s_lcd_romy
                RET

lcd_ram_s34:    MOV     DPTR,#men_abrd          ; abonado abreviado
                LCALL   s_lcd_romy
                MOV     A,@R1
                LCALL   s_lcd_wr
                RET
```

```
;&&     ==============================
;       s_lcd_rom
;       s_lcd_romr1
;       s_lcd_romrm
;       display_escribir de rom (con retardo ...)
;
;       Escribe en el display el mensaje hallado en ROM direccionado por DPTR.
;       El mensaje consta de dos lineas, terminadas cada una en "$".
;       Segun la llamada, proporciona un retardo para poder leer el mensaje:
;              f_rm    f_r1     retardo
;               0       0          0 ms
;               0       1       1000 ms
;               1       0       2560 ms
;
;       FLAG    f_l3, f_rm, f_r1
;       IN      DPTR
;       ==============================
s_lcd_romrm:    SETB    f_rm
                CLR     f_r1
                AJMP    lcd_rom_s2

s_lcd_romr1:    CLR     f_rm
                SETB    f_r1
                AJMP    lcd_rom_s2

s_lcd_rom:      CLR     f_rm
                CLR     f_r1

lcd_rom_s2:     LCALL   s_lcd_clr
                MOV     lcd_pun,#0FFH

lcd_rom_s4:     INC     lcd_pun
                MOV     A,lcd_pun
                MOVC    A,@A+DPTR
                CJNE    A,#24H,lcd_rom_s6       ; compara con $
                JBC     f_l3,lcd_rom_s8
                SETB    f_l3
                MOV     A,#0C0H                 ; cursor al inicio de 2da fila
                LCALL   s_lcd_wrd
                AJMP    lcd_rom_s4

lcd_rom_s6:     LCALL   s_lcd_wr
                AJMP    lcd_rom_s4

lcd_rom_s8:     JB      f_rm,lcd_rom_s10
                JB      f_r1,lcd_rom_s12
                RET

lcd_rom_s10:    LCALL   s_ret_max               ; retardo = 2560 ms
                RET

lcd_rom_s12:    LCALL   s_ret_1000              ; retardo = 1000 ms
                RET
```

```
;&&     ==============================
;       s_lcd_romx
;       s_lcd_romy
;       display_escribir de rom...
;
;       s_lcd_romx: escribe en el display, desde la localidad indicada en A,
;               el mensaje hallado en ROM y direccionado por DPTR hasta
;               encontrar "$".
;       s_lcd_romy: escribe en el display, a partir de la posicion del cursor,
;               el mensaje hallado en ROM y direccionado por DPTR hasta
;               encontrar "$".
;
;       IN      A, DPTR
;       ==============================
s_lcd_romx:     LCALL   s_lcd_wrd               ; posicionar el cursor en lcd_dat

s_lcd_romy:     MOV     lcd_pun,#0FFH
lcd_romx_s2:    INC     lcd_pun                 ; escribir hasta hallar $
                MOV     A,lcd_pun
                MOVC    A,@A+DPTR
                CJNE    A,#24H,lcd_romx_s4
                RET

lcd_romx_s4:    LCALL   s_lcd_wr
                AJMP    lcd_romx_s2



;&&     ==============================
;       s_lcd_wrd
;       s_lcd_wrtx
;       s_lcd_wrti
;       s_lcd_wr
;       display_escribir un caracter
;
;       s_lcd_wrd: setea f_l1 para que s_lcd_wr trate el contenido de A
;               como un dato de inicializacion (p_lcd_ini) o una
;               direccion (en posicionamiento del cursor).
;       s_lcd_wrtx: eco al display de la tecla presionada (teclado externo),
;               debuelve en A el ASCII de la tecla.
;       s_lcd_wrti: eco al display de la tecla presionada (teclado interno),
;               debuelve en A el ASCII de la tecla; el teclado esta
;               determinado por f_tecla1, f_tecla3.
;       s_lcd_wr: envia al display el contenido de A, sin alterarlo.
;
;       IN      f_tecla1, f_tecla3
;       FLAG    f_l1, f_l2
;       ==============================
s_lcd_wrd:      SETB    f_l1
                AJMP    s_lcd_wr

s_lcd_wrti:     SETB    f_ti
                AJMP    lcd_wrt_s0

s_lcd_wrtx:     CLR     f_ti
lcd_wrt_s0:     PUSH    DPH
                PUSH    DPL
                MOV     DPTR,#ramx3             ; obtener valor SSI de la tecla
                MOVX    A,@DPTR
                ANL     A,#0FH
                DEC     A
                PUSH    A

                JB      f_ti,lcd_wrt_s2
                MOV     DPTR,#tab_teclax        ; teclado externo activo
                AJMP    lcd_wrt_s8

lcd_wrt_s2:     JNB     f_tecla1,lcd_wrt_s4
                LCALL   s_lcd_clr
                MOV     DPTR,#tab_tecla1        ; teclado activo 1
                AJMP    lcd_wrt_s8

lcd_wrt_s4:     JNB     f_tecla3,lcd_wrt_s6
```

```
                LCALL   s_lcd_clr
                MOV     DPTR,#tab_tecla3        ; teclado activo 3
                AJMP    lcd_wrt_s8

lcd_wrt_s6:     MOV     DPTR,#tab_tecla5        ; teclado activo 5

lcd_wrt_s8:     POP     A
                MOVC    A,@A+DPTR
                AJMP    lcd_wr_s0

;       ********        escritura en el display
s_lcd_wr:       PUSH    DPH                     ; f_l1=1 => direcciones, control
                PUSH    DPL                     ; f_l1=0 => caracter

lcd_wr_s0:      MOV     lcd_dat,A               ; salvar el dato a escribir
                MOV     DPTR,#ramx1
                SWAP    A
lcd_wr_s2:      ANL     A,#0FH
                JB      f_l1,lcd_wr_s4
                SETB    dis_rs

lcd_wr_s4:      MOVX    @DPTR,A
                SETB    dis_e
                MOVX    @DPTR,A

                MOV     TL0,#9AH                ; retardo = 1ms
                MOV     TH0,#0FDH
                SETB    TR0                     ; timer 0 habilitado
ret_1:          NOP
                JNB     TF0,ret_1
                CLR     TF0                     ; borrar bandera de overflow
                CLR     TR0                     ; timer 0 deshabilitado

                CLR     dis_e
                MOVX    @DPTR,A

                JBC     f_l2,lcd_wr_s6
                SETB    f_l2
                MOV     A,lcd_dat
                AJMP    lcd_wr_s2
lcd_wr_s6:      CLR     f_l1
                MOV     A,lcd_dat               ; restituir acumulador
                POP     DPL
                POP     DPH
                RET
```

```
;&&     ********************************************************
;                       LINEA: MARCADO, RELES
;       ********************************************************

;       ==============================
;       s_mar1
;       marcar automaticamente un numero telefonico en lt1
;
;       Una vez recibida la invitacion a marcar, marca en tonos o pulsos,
;       dependiendo del valor de f_pt1, el numero telefonico hallado en el
;       banco del numero, con una cadencia de:
;               f_pt
;               0 (tonos)       tono:  200ms
;                               pausa:  50ms
;               1 (pulsos)      cierre:              400ms
;                               apertura:            600ms
;                               tiempo entre digitos: 700ms
;
;       Una vez marcado el numero se carga en el banco formado por las
;       localidades hh1, mm1 y ss1 la hora de inicio de la llamada y se set
;       la bandera f_cro1 para indicar al procedimiento p_reloj de debe
;       cronometrar a la linea 1.
;       La unica forma de interrumpir el marcado es desocupando la linea.
;
;       IN      banco del numero
;       OUT     f_att1
;       ==============================
s_mar1:         LCALL   s_lcd_cln
                LCALL   s_r_dr1                 ; conexion lector dtmf

mar1_s0:        NOP                             ; espera invitacion a marcar
                JB      ix_lt1,mar1_s24         ; lt1 ocupada?
                JNB     ix_425,mar1_s0

                MOV     R5,#32H                 ; tono de invitacion debe
mar1_s2:        LCALL   s_ret_10                ; estar presente 500ms
                JNB     ix_425,mar1_s0
                JB      ix_dtmf,mar1_s4
                DJNZ    R5,mar1_s2
                AJMP    mar1_s8

mar1_s4:        NOP
                JB      ix_dtmf,mar1_s4
mar1_s6:        NOP
                JB      ix_425,mar1_s6
                AJMP    mar1_s0

mar1_s8:        JNB     f_pt1,mar1_s10          ; si linea de pulsos
                LCALL   c_r_dr1                 ; desconecta lector dtmf

mar1_s10:       LCALL   s_ret_100               ; 100ms antes de marcar
                MOV     R1,#numero-1
                MOV     R2,#0EH                 ; 14 numeros
mar1_s12:       JB      ix_lt1,mar1_s24         ; lt1 ocupada?
                INC     R1
                MOV     A,@R1
                LCALL   s_lcd_wr

                CJNE    A,#3AH,mar1_s14         ; numero con menos de
                AJMP    mar1_s22                ; 14 digitos

mar1_s14:       ANL     A,#0FH
                JB      f_pt1,mar1_s30

;       ********        lt1 de tonos (marcar en tonos)
                LCALL   s_mar_dtmf
                DJNZ    R2,mar1_s12

mar1_s22:       MOV     R0,#ss1                 ; marcado el num completo
                LCALL   s_cha_hex               ; carga la hora de inicio de
                SETB    f_cro1                  ; la llout1
                SETB    f_att1                  ; lt1 ya atendida
mar1_s24:       LCALL   c_r_dr1                 ; desconexion lector dtmf
                RET
```

```
;       ********        lt1 de pulsos (marcar en pulsos)
mar1_s30:       JZ      mar1_s36
                MOV     R3,A

mar1_s32:       LCALL   s_r_lt1
                LCALL   s_ret_20
                LCALL   c_r_lt1
                DJNZ    R3,mar1_s32
                LCALL   s_ret_700               ; .7seg entre numeros
                DJNZ    R2,mar1_s12
                AJMP    mar1_s22

mar1_s36:       MOV     R3,#0AH
                AJMP    mar1_s32



;&&     ==============================
;       s_mar2
;       marcar automaticamente un numero telefonico en lt2
;       ==============================
s_mar2:         LCALL   s_lcd_cln
                LCALL   s_r_dr2                 ; conexion lector dtmf

mar2_s0:        NOP                             ; espera invitacion a marcar
                JB      ix_lt2,mar2_s24         ; lt2 ocupada?
                JNB     ix_425,mar2_s0

                MOV     R5,#32H                 ; tono de invitacion debe
mar2_s2:        LCALL   s_ret_10                ; estar presente 500ms
                JNB     ix_425,mar2_s0
                JB      ix_dtmf,mar2_s4
                DJNZ    R5,mar2_s2
                AJMP    mar2_s8

mar2_s4:        NOP
                JB      ix_dtmf,mar2_s4
mar2_s6:        NOP
                JB      ix_425,mar2_s6
                AJMP    mar2_s0

mar2_s8:        JNB     f_pt2,mar2_s10          ; si linea de pulsos
                LCALL   c_r_dr2                 ; desconecta lector dtmf

mar2_s10:       LCALL   s_ret_100               ; 100ms antes de marcar
                MOV     R1,#numero-1
                MOV     R2,#0EH                 ; 14 numeros
mar2_s12:       JB      ix_lt2,mar2_s24         ; lt2 ocupada?
                INC     R1
                MOV     A,@R1
                LCALL   s_lcd_wr

                CJNE    A,#3AH,mar2_s14         ; numero con menos de
                AJMP    mar2_s22                ; 14 digitos

mar2_s14:       ANL     A,#0FH
                JB      f_pt2,mar2_s30

;       ********        lt2 de tonos (marcar en tonos)
                LCALL   s_mar_dtmf
                DJNZ    R2,mar2_s12

mar2_s22:       MOV     R0,#ss2                 ; marcado el num completo
                LCALL   s_cha_hex               ; hora inicio de llout2
                SETB    f_cro2                  ; cronometrar llout2
                SETB    f_att2                  ; lt2 ya atendida
mar2_s24:       LCALL   c_r_dr2                 ; desconexion lector dtmf
                RET

;       ********        lt2 de pulsos (marcar en pulsos)
mar2_s30:       JZ      mar2_s36
                MOV     R3,A
```

```
mar2_s32:       LCALL   s_r_lt2
                LCALL   s_ret_20
                LCALL   c_r_lt2
                DJNZ    R3,mar2_s32
                LCALL   s_ret_700               ; .7seg entre numeros
                DJNZ    R2,mar2_s12
                AJMP    mar2_s22

mar2_s36:       MOV     R3,#0AH
                AJMP    mar2_s32



;&      ==============================
;       s_mar_dtmf
;       marcar tonos
;
;       marca el numero definido por A en tonos
;
;       IN      A
;       ==============================
s_mar_dtmf:     MOV     DPTR,#tab_dw            ; tono 200ms
                MOVC    A,@A+DPTR
                MOV     DPTR,#ramx4
                MOVX    @DPTR,A
                LCALL   s_ret_200
                MOV     A,#0F0H                 ; marcador ocio 50ms
                MOVX    @DPTR,A
                LCALL   s_ret_50
                RET



;       ==============================
;       s_cha_bcd
;       s_cha_hex
;       cargar hora actual
;
;       Carga en el banco apuntado por R0 el segundo, minuto y hora del sistema
;       en formato bcd (s_cha_bcd) o en formato hexadecimal (s_cha_hex).
;
;       IN      R0
;       FLAG    f_hex
;       ==============================
s_cha_bcd:      CLR     f_hex
                AJMP    cha_s0

s_cha_hex:      SETB    f_hex

cha_s0:         MOV     DPTR,#c_seg
                MOV     R1,#03H

cha_s2:         MOVX    A,@DPTR
                MOV     @R0,A
                JNB     f_hex,cha_s4
                ANL     A,#0F0H                 ; paso de BCD a hexadecimal
                SWAP    A
                MOV     B,#0AH
                MUL     AB
                XCH     A,@R0
                ANL     A,#0FH
                ADD     A,@R0
                MOV     @R0,A
cha_s4:         INC     DPTR
                INC     R0
                DJNZ    R1,cha_s2
                RET
```

```
;&&     ==============================
;       c_r_..
;       s_r_..
;       s_rele
;       s_relesr
;       clr / set el rele...
;
;       c_r_...: pone 0 en el bit indicado del banco de reles y lo copia en
;           la localidad de RAM externa 0A000H (ramx5).
;       s_r_...: pone 1 en el bit indicado del banco del rele y copia el contenido
;           del banco de reles en la localidad de RAM externa 0A000H (ramx5).
;       s_rele: copia el contenido del banco de reles en la localidad de RAM
;           externa 0A000H (ramx5), RETARDO = 40ms.
;       s_relesr: copia el contenido del banco de reles en la localidad de RAM
;           externa 0A000H (ramx5).
;       ==============================
c_r_lt1:        CLR     r_lt1           ; lt1 sin bloqueo
                AJMP    s_rele

s_r_lt1:        SETB    r_lt1           ; lt1 con bloqueo
                AJMP    s_rele

c_r_dr1:        CLR     r_dr1           ; desconectado wr/rd tonos para lt1
                AJMP    s_rele

s_r_dr1:        SETB    r_dr1           ; conectado wr/rd tonos para lt1
                AJMP    s_rele

c_r_lt2:        CLR     r_lt2           ; lt2 sin bloqueo
                AJMP    s_rele

s_r_lt2:        SETB    r_lt2           ; lt2 con bloqueo
                AJMP    s_rele

c_r_dr2:        CLR     r_dr2           ; desconectado wr/rd tonos para lt2
                AJMP    s_rele

s_r_dr2:        SETB    r_dr2           ; conectado wr/rd tonos para lt2

s_rele:         LCALL   s_ret_20
                MOV     DPTR,#ramx5
                MOV     A,ban_rele
                MOVX    @DPTR,A
                LCALL   s_ret_20
                RET

c_r_flt1:       CLR     r_flt1          ; filtro para lt1 habilitado
                AJMP    s_relesr

s_r_flt1:       SETB    r_flt1          ; filtro para lt1 deshabilitado
                AJMP    s_relesr

c_r_llin1:      CLR     r_llin1         ; senal de timbre detectada en lt1
                AJMP    s_relesr

s_r_llin1:      SETB    r_llin1         ; no existe senal de timbre en lt1
                AJMP    s_relesr

c_r_flt2:       CLR     r_flt2          ; filtro para lt2 habilitado
                AJMP    s_relesr

s_r_flt2:       SETB    r_flt2          ; filtro para lt2 deshabilidado
                AJMP    s_relesr

c_r_llin2:      CLR     r_llin2         ; senal de timbre detectada en lt2
                AJMP    s_relesr

s_r_llin2:      SETB    r_llin2         ; no existe senal de timbre en lt2

s_relesr:       PUSH    DPH
                PUSH    DPL
                PUSH    A
                MOV     DPTR,#ramx5
                MOV     A,ban_rele
```

```
                MOVX    @DPTR,A
                POP     A
                POP     DPL
                POP     DPH
                RET


;&      ============================
;       c_rd_pls
;       leer pulsos
;
;       determina que numero se ha marcado en pulsos, lo escribe en el
;       display y devuelve su ASCII en A
;
;       OUT     A
;       ============================
s_rd_pls:       CLR     A                       ; contador numero marcado
                MOV     R1,#00H                 ; contador tiempo transcurrido

rd_pls_s2:      LCALL   s_ret_10
                INC     R1

                JB      f_lt2att,rd_pls_s4      ; definir la linea atendida
                JNB     ix_lt1,rd_pls_s10
                AJMP    rd_pls_s6

rd_pls_s4:      JNB     ix_lt2,rd_pls_s10
rd_pls_s6:      CJNE    R1,#1EH,rd_pls_s2       ; si alto dura mas de 300ms
                RET                             ; validar linea desocupada

rd_pls_s10:     INC     A                       ; bajo en menos de 300ms
                MOV     R1,#00H
rd_pls_s12:     LCALL   s_ret_10
                INC     R1

                JB      f_lt2att,rd_pls_s14     ; definir la linea atendida
                JB      ix_lt1,rd_pls_s2
                AJMP    rd_pls_s16

rd_pls_s14:     JB      ix_lt2,rd_pls_s2
rd_pls_s16:     CJNE    R1,#32H,rd_pls_s12      ; si bajo dura mas de 500ms
                CJNE    A,#0AH,rd_pls_s20       ; validar numero marcado
                CLR     A
rd_pls_s20:     ORL     A,#30H
                LCALL   s_lcd_wr
                RET
```

```
;&&     ************************************************************
;                        RELOJ
;       ************************************************************

;       ==============================
;       s_feho
;       fecha hora
;
;       Sacar al display la fecha y hora del sistema en el siguiente formato:
;       fil1:           dia de la semana, dia del mes, mes
;       fil2:           hora, minuto, segundo.
;       ==============================
s_feho:         MOV     A,#80H                  ; cursor al inicio
                MOV     DPTR,#men_clrrel
                INC     DPTR
                LCALL   s_lcd_romx

                MOV     DPTR,#c_dis             ; dia de la semana
                MOVX    A,@DPTR
                MOV     DPTR,#tab_dis           ; mostrar nombre
                LCALL   s_lcd_ck1

                MOV     DPTR,#c_dim             ; dia del mes
                LCALL   s_lcd_bcdd

                MOV     DPTR,#c_mes             ; mes
                MOVX    A,@DPTR                 ; paso de BCD a HEX
                JNB     d_dim,feho_s12
                CLR     d_dim
                ADD     A,#0AH
feho_s12:       MOV     DPTR,#tab_mes           ; mostrar nombre
                LCALL   s_lcd_ck1

                MOV     A,#0C0H                 ; cursor al inicio de fila 2
                MOV     DPTR,#men_clrrel
                LCALL   s_lcd_romx

                MOV     R0,#ssd                 ; mostrar hora
                LCALL   s_cha_bcd
                LCALL   s_lcd_hms
                RET



;&&     ==============================
;       s_crono
;       cronometro
;
;       El tiempo mostrado en el display se lo obtiene restando la hora de
;       inicio de la llamada (cargada en el banco apropiado por las subrutinas
;       s_mar1 y s_mar2) de la hora del sistema.
;               f_cro1  f_cro2  funcion
;               1       0       cronometrar lt1
;               0       1       cronometrar lt2
;               1       1       cronometrar lt1 y lt2
;
;       IN      f_cro1, f_cro2
;       ==============================
s_crono:        MOV     R0,#ssa                 ; cargar hora actual
                LCALL   s_cha_hex

                JNB     f_cro1,crono_s2
                MOV     R1,#ss1
                AJMP    crono_s4

crono_s2:       MOV     R1,#ss2
crono_s4:       MOV     A,ssa                   ; calcular segundos
                CLR     C
                SUBB    A,@R1
                INC     R1
                JC      crono_s6
                MOV     ssd,A
                AJMP    crono_s12
```

```
crono_s6:       ADD     A,#3CH                  ; sumar 60 a los segundos
                MOV     ssd,A                   ; y decrementar un minuto
                MOV     A,mma
                JZ      crono_s8
                DEC     mma
                AJMP    crono_s12

crono_s8:       MOV     mma,#3BH                ; cargar 59 en los minutos
                MOV     A,hha                   ; y decrementar una hora
                JZ      crono_s10
                DEC     hha
                AJMP    crono_s12

crono_s10:      MOV     hha,#17H                ; cargar 23 en las horas

crono_s12:      MOV     A,mma                   ; calcular minutos
                CLR     C
                SUBB    A,@R1
                INC     R1
                JC      crono_s14
                MOV     mmd,A
                AJMP    crono_s20

crono_s14:      ADD     A,#3CH                  ; sumar 60 a los minutos
                MOV     mmd,A                   ; y decrementar una hora
                MOV     A,hha
                JZ      crono_s16
                DEC     hha
                AJMP    crono_s20

crono_s16:      MOV     hha,#17H                ; cargar 23 en las horas

crono_s20:      MOV     A,hha                   ; calcular horas
                CLR     C
                SUBB    A,@R1
                JC      crono_s22
                MOV     hhd,A
                AJMP    crono_s30

crono_s22:      ADD     A,#18H
                MOV     hhd,A

crono_s30:      MOV     DPTR,#tab_hex_bcd       ; paso de hexadec a bcd
                MOV     R0,#ssd
                MOV     R2,#03H

crono_s32:      CLR     A
                XCHD    A,@R0
                CLR     AC                      ; paso a BCD el lsn
                DA      A
                XCH     A,@R0
                SWAP    A                       ; paso a BCD el msn
                MOVC    A,@A+DPTR
                ADD     A,@R0
                DA      A
                MOV     @R0,A
                INC     R0
                DJNZ    R2,crono_s32

;       si el valor de R1 = hh1  => se ha cronometrado lt1
;       si el valor de R1 <> hh1 => se ha cronometrado lt1

                CJNE    R1,#hh1,crono_s36
                MOV     A,#80H                  ; cronometra lt1
                MOV     DPTR,#men_crlt1
                LCALL   s_lcd_romx
                LCALL   s_lcd_hms

                JB      f_cro2,crono_s2
                MOV     A,#0C0H
                MOV     DPTR,#men_clrfil
                LCALL   s_lcd_romx
                RET
```

```
crono_s36:      JB      f_cro1,crono_s38        ; cronometra lt2
                MOV     A,#80H
                MOV     DPTR,#men_clrfil
                LCALL   s_lcd_romx

crono_s38:      MOV     A,#0C0H
                MOV     DPTR,#men_crlt2
                LCALL   s_lcd_romx
                LCALL   s_lcd_hms
                RET
```

```
;&&     ************************************************************
;                       TECLADO
;       ************************************************************

;       ==============================
;       s_t_alfn
;       s_t_alf
;       teclado alfabetico y numerico
;
;       s_t_alfn: reconoce teclado para modo navegacion y alfabetico.
;       s_t_alf: reconoce teclado solo para modo alfabetico.
;
;     - En modo navegacion incrementa o decrementa en uno la direccion del
;       abonado actual, segun el caso; indica ademas si se ha llegado al
;       inicio (BOF) o al fin (EOF) del directorio y de ser asi apunta al
;       primero o al ultimo abonado.
;     - En modo alfabetico, reconoce la tecla y las veces consecutivas que
;       esta ha sido presionada, de acuerdo a esto, devuelve en A el ASCII
;       del grupo escogido.
;
;       IN      A, tec_pre
;       OUT     f_alf, dir_pun, f_dp, A
;       FLAG    f_tpos
;       ==============================
;       ********        navegacion: 1, 0
s_t_alfn:       CLR     f_alf

                CJNE    A,#20H,t_alfn_s10       ; [1] anterior
                LCALL   s_dec_dp                ; decrementar puntero
                MOV     A,dir_pun
                JNZ     t_alfn_s4
                JB      f_dp,t_alfn_s4
                LCALL   s_inc_dp
                MOV     DPTR,#men_bof           ; es BOF
t_alfn_s2:      LCALL   s_lcd_rom
                LCALL   s_beep
                LCALL   s_ret_250
t_alfn_s4:      RET

t_alfn_s10:     CJNE    A,#30H,s_t_alf          ; [0] siguiente
                LCALL   s_inc_dp                ; incrementar el puntero
                LCALL   s_add_ab
                MOVX    A,@DPTR
                CJNE    A,#car_findir,t_alfn_s4
                LCALL   s_dec_dp
                MOV     DPTR,#men_eof           ; es EOF
                AJMP    t_alfn_s2

;       ********        alfabeto: 2 - 9
s_t_alf:        SETB    f_alf
                CJNE    A,#50H,t_alf_s2         ; teclas con 4 posibilidades
                AJMP    t_alf_s4
t_alf_s2:       CJNE    A,#57H,t_alf_s10
t_alf_s4:       SETB    f_tpos

t_alf_s10:      CJNE    A,tec_pre,t_alf_s18
                JBC     f_tpos,t_alf_s12
                CJNE    R3,#02H,t_alf_s16       ; tecla con 3 posibilidades
                AJMP    t_alf_s14

t_alf_s12:      CJNE    R3,#03H,t_alf_s16       ; R3 contador del numero de
t_alf_s14:      MOV     R3,#00H                 ; veces que la misma tecla
                RET                             ; se ha presionado

t_alf_s16:      INC     R3
                ADD     A,R3
                RET

t_alf_s18:      MOV     tec_pre,A
                MOV     R3,#00H
                RET
```

```
;&&     ==============================
;       s_tecla0
;       teclado 0
;
;       Lectura teclado 0 sin eco al display; devuelve en A el ASCII
;       correspondiente de la tecla una vez liberada.
;
;       OUT     A
;       ==============================
s_tecla0:
tecla0_s0:      NOP                             ; esperar tecla presionada
                JB      f_lt2att,tecla0_s2
                JNB     ix_tel1,tecla0_s10      ; lt1
                AJMP    tecla0_s4

tecla0_s2:      JNB     ix_tel2,tecla0_s10      ; lt2
tecla0_s4:      JNB     ix_dtmf,tecla0_s0       ; existe DV

                MOV     DPTR,#ramx3             ; obtener valor SSI de la tecla
                MOVX    A,@DPTR
                ANL     A,#0FH
                DEC     A
                MOV     DPTR,#tab_tecla1
                MOVC    A,@A+DPTR

tecla0_s6:      NOP                             ; esperar tecla liberada
                JB      ix_dtmf,tecla0_s6
                RET

tecla0_s10:     MOV     R6,#00H                 ; validar telefono colgado
tecla0_s12:     LCALL   s_ret_10                ; si este pasa asi por mas
                INC     R6                      ; de 0.3 seg

                JB      f_lt2att,tecla0_s14
                JB      ix_tel1,tecla0_s0       ; lt1
                CJNE    R6,#1EH,tecla0_s12
                RET

tecla0_s14:     JB      ix_tel2,tecla0_s0       ; lt2
                CJNE    R6,#1EH,tecla0_s12
                RET



;&&     ==============================
;       s_teclan
;       teclado n (1, 3 xor 5)
;
;       Lectura teclado n con eco al display; devuelve en A el ASCII
;       correspondiente a la tecla una vez liberada.
;
;       FLAG    f_tecla1, f_tecla3
;       OUT     A
;       ==============================
s_tecla5:       CLR     f_tecla1                ; f_tecla3 f_tecla1 teclado
                CLR     f_tecla3                ;    0        1        1
                AJMP    tecla_s0                ;    1        0        3
                                                ;    0        0        5
s_tecla3:       CLR     f_tecla1
                SETB    f_tecla3
                AJMP    tecla_s0

s_tecla1:       SETB    f_tecla1
                CLR     f_tecla3

tecla_s0:       NOP                             ; esperar tecla presionada
                JB      f_lt2att,tecla_s2
                JNB     ix_tel1,tecla_s20       ; lt1
                AJMP    tecla_s4

tecla_s2:       JNB     ix_tel2,tecla_s20       ; lt2
tecla_s4:       JNB     ix_dtmf,tecla_s0        ; existe DV
```

```
                LCALL   s_lcd_wrti

tecla_s10:      NOP
                JB      ix_dtmf,tecla_s10       ; espera tecla liberada
                RET

tecla_s20:      MOV     R6,#00H                 ; validar telefono colgado
tecla_s22:      LCALL   s_ret_10                ; si este pasa asi por mas
                INC     R6                      ; de 0.3 seg

                JB      f_lt2att,tecla_s24
                JB      ix_tel1,tecla_s0        ; lt1
                CJNE    R6,#1EH,tecla_s22
                RET

tecla_s24:      JB      ix_tel2,tecla_s0        ; lt2
                CJNE    R6,#1EH,tecla_s22
                RET



;&      ==============================
;       s_tpn
;       tres primeros numeros
;
;       Determina si los tres primeros numero pulsados corresponden a:
;               numeros pulsados                bandera seteada
;               cod de activado: 0*             f_on
;               LDN: 105 (oper) / 0x (1<=x<=8)  f_ldn
;               LDI: 116 (oper) / 00            f_ldi
;
;       REGISTROS       R0, contador
;       IN              A (ASCII de la tecla)
;       OUT             f_on, f_ldn, f_ldi
;       FLAG            f_k2
;       ==============================
s_tpn:          CJNE    R0,#03H,tpn_s10
                CJNE    A,#30H,tpn_s2
                CLR     f_k2                    ; primera tecla pres = 0
                RET

tpn_s2:         CJNE    A,#31H,tpn_s4
                SETB    f_k2                    ; primera tecla pres = 1
                RET

tpn_s4:         SETB    f_k1
                RET

;       ********        segunda tecla presionada
tpn_s10:        CJNE    R0,#02H,tpn_s30
                JB      f_k2,tpn_s20
                SETB    f_k1
                CJNE    A,#2AH,tpn_s12          ; cod acceso: 0*
                SETB    f_on
                RET

tpn_s12:        CJNE    A,#30H,tpn_s14          ; LDI: 00
                SETB    f_ldi
                RET

tpn_s14:        CJNE    A,#39H,tpn_s16          ; celular: 09 sin bloqueo
                SETB    f_cel
                RET

tpn_s16:        CJNE    A,#23H,tpn_s18          ; 0# sin bloqueo
                RET

tpn_s18:        SETB    f_ldn                   ; LDN: 0x (1<= x <=8)
                RET

tpn_s20:        CJNE    A,#30H,tpn_s22
                CLR     f_k2
                RET
```

```
;&&     ********************************************************
;                       MENSAJES
;      ********************************************************

men_abr:        DB      ' SEL   abr asig$'
                DB      '       $'
men_abrd:       DB      'ab$'
men_abrna:      DB      '  abreviado(s)$'
                DB      ' NO ASIGNADO(S)$'
men_abrok:      DB      '  abreviado OK$'
                DB      '$'
men_agre:       DB      'gregar$'
men_autom:      DB      '   AUTOMATICO$'
men_blo:        DB      ' BLOQUEO   $'
men_bof:        DB      '  BOF$'
                DB      '$'
men_cargaok:    DB      ' CARGA COMPLETA$'
                DB      '$'
men_cel:        DB      ' celular$'
men_clrall:     DB      '    INGRESAR$'
                DB      '   NUEVAMENTE$'
men_clrdir:     DB      '    CLR  DIR$'
                DB      '$'
men_clrfil:     DB      '                $'
men_clrfin:     DB      '    clr  dir$'
                DB      '    COMPLETO$'
men_clrhotl:    DB      'hl $'
men_clrmabr:    DB      'ab $'
men_clrrel:     DB      '       $'
men_cnf:        DB      '1pin  2pint  eNE$'
                DB      '$'
men_crlt1:      DB      ' lt1:  $'
men_crlt2:      DB      ' lt2:  $'
men_dirllep:    DB      'directorio LLENO$'
men_dirvac:     DB      '   directorio$'
                DB      '     VACIO$'
men_dirvacp:    DB      'directorio VACIO$'
men_elim:       DB      'liminar$'
men_elimok:     DB      '  ELIMINACION$'
                DB      '   COMPLETA$'
men_eof:        DB      '  EOF$'
                DB      '$'
men_eqac:       DB      '     equipo$'
                DB      '     ACTIVO$'
men_er:         DB      ' er:$'
men_ext:        DB      '.rnd$'
men_frx:        DB      '  recepcion OK$'
                DB      '$'
men_ftx:        DB      ' transmision OK$'
                DB      '$'
men_hld:        DB      '    hot def$'
                DB      '  $'
men_hlnd:       DB      ' ¥¥¥¥$'
men_hl1d:       DB      'hl1$'
men_hl2d:       DB      'hl2$'
men_hotl:       DB      'ot line$'
men_hotoff:     DB      '$'
                DB      ' hot line:  OFF$'
men_hoton:      DB      ' hot line:  ON$'
                DB      '$'
men_infp:       DB      '  informacion$'
                DB      '   protegida$'
men_infpc:      DB      ' inf  protegida $'
men_ldi:        DB      '     ldi$'
men_ldn:        DB      '     ldn$'
men_line:       DB      'inea - Equipo$'
men_llin1:      DB      '    ll_in  1$'
                DB      '$'
men_llin2:      DB      '    ll_in  2$'
                DB      '$'
men_loc:        DB      '   local$'
men_lt1:        DB      ' LT_1$'
                DB      '$'
men_lt2:        DB      ' LT_2$'
```

```
                DB      '$'
men_mabr:       DB      'arc abreviada$'
men_nf1:        DB      '  "$'
men_nf2:        DB      '"  GRUPO    $'
men_nf3:        DB      '       VACIO    $'
men_ok:         DB      ' OK$'
men_pass:       DB      'assword$'
men_prog:       DB      '      modo$'
                DB      '  PROGRAMACION$'
men_prplt1:     DB      '  programacion$'
                DB      ' solo desde LT1$'
men_rx:         DB      '      modo$'
                DB      '   RECEPCION$'
men_ser:        DB      '      modo$'
                DB      '     SERIAL$'
men_tiem:       DB      'iempo$'
men_tiemin:     DB      'MeDmDshhmm$'
                DB      '$'
men_tx:         DB      '      modo$'
                DB      '  TRANSMISION$'


;&      ********************************************************
;                       TABLAS
;       ********************************************************

tab_cnf:        DB      0C1H,0C2H,0C3H,0C7H,0C8H,0C9H,0CAH,0CEH
tab_dis:        DB      'lun mar mie jue vie sab dom '
tab_dw:         DB      72H             ; 0
                DB      0E1H,0E2H,0E4H  ; 1 2 3
                DB      0D1H,0D2H,0D4H  ; 4 5 6
                DB      0B1H,0B2H,0B4H  ; 7 8 9
                DB      0F0H            ; marcador ociosos
tab_ini:        DB      33H,32H
                DB      28H             ; interfaz 4 bits, dual line, 5x7
                DB      0CH             ; display on, cursor off, blink off
                DB      06H             ; incrementar, not shift
tab_hex_bcd:    DB      00H,16H,32H,48H
tab_hex_dec:    DB      56H,02H         ; ocho
                DB      28H,01H         ; siete
                DB      64H,00H         ; seis
                DB      32H,00H         ; cinco
                DB      16H,00H         ; cuatro
                DB      08H,00H         ; tres
                DB      04H,00H         ; dos
                DB      02H,00H         ; uno
                DB      01H,00H         ; cero
tab_mes:        DB      ' ENE FEB MAR ABR MAY JUN JUL AGO SEP OCT NOV DIC'
tab_pls:        DB      31H,32H,33H,34H,35H,36H,37H,38H,39H,30H
tab_teclax:     DB      '1234567890*#'
tab_tecla1:     DB      ' ADGJMPTW0*#'
tab_tecla3:     DB      ' AEHLMPT  *ï'
tab_tecla5:     DB      '1234567890 :'


;       ********************************************************


men_ini:        DB      '... NOSTROMO ...$'
                DB      '*  team Ronda  *$'
men_nomeq:      DB      '.... ADITEL ....$'
                DB      ' **   eq96   **$'
men_ver:        DB      ' ver:  P6042908$'
                DB      '      96 JUN 01$'

                END                                     ; fin del ensamblado
```

anexo 2.2

Listado del programa para el computador personal

```
'         ================================================
'         PROGRAMA PARA ADMINISTRAR DIRECTORIOS TELEFONICOS
'                          ADITEL.BAS
'                           96/05/06
'
'              - agregar abonados
'              - borrar abonados
'              - consutar listado
'              - definir tipo de abonado
'              - imprimir listado
'              - transmitir/recivir listado del equipo
'              - administracion de archivos
'              - tratamiento de errores
'         ================================================


DECLARE SUB abrir (nomarchivo$)
DECLARE SUB cerrar (nomarchivo$)
DECLARE SUB direstado ()
DECLARE SUB direcyarchi (nomdirectorio$, totarchivos%, listarchivos$())

DECLARE SUB menu (nomarchivo$, menuinici%, opcion$)
DECLARE SUB menupri (numenu%, habilmp%, menuinici%)
DECLARE SUB menupop (numenu%, sileerteclado%, opcion$, finmen%, menuinici%)

DECLARE SUB nuevoabonado (nomarchivo$)
DECLARE SUB borrar (nomarchivo$)
DECLARE SUB consultar (recno%)
DECLARE SUB consultarchivo (opcion$, archinum%, filactiva%, totarchivos%, listarchivos$())
DECLARE SUB definir (nomarchivo$)
DECLARE SUB imprimir (nomarchivo$)
DECLARE SUB recibir (nomarchivo$, cadenarx$)
DECLARE SUB transmitir (nomarchivo$)

DECLARE SUB ingcadena (cadena$, tipo$, escape%)
DECLARE SUB ingnuevodir (nuevodirectorio$, escape%)
DECLARE SUB browse (recno%, filactiva%, numabonados%)
DECLARE SUB browsearchi (archinum%, filactiva%, totarchivos%, listarchivos$())

DECLARE SUB about ()
DECLARE SUB menayuda (numayuda%)
DECLARE SUB manual ()

DECLARE SUB flechabr (filactiva%)
DECLARE SUB flechamp (numenu%, nupop%)
DECLARE SUB flecharchi (filactiva%, totarchivos%)
DECLARE SUB leerteclado (tecla$)

DECLARE SUB cuadro (tipcuad%, fili%, coli%, filf%, colf%)
DECLARE SUB escrimen (tipmen%, fila%, mensaje$)
DECLARE SUB escrirot (tiprot%, mensaje$, opcion%)


CONST tiparch = "A"                 ' cadena tipo archivo
CONST tipnoma = "N"                 ' cadena tipo nombre del abonado
CONST tiptela = "T"                 ' cadena tipo telefono del abonado
CONST longarchivo = 32              ' longitud del registro

CONST numpuerto = 1                 ' numero del puerto serial
CONST numarchivo = 2                ' numero del archivo de trabajo
CONST numartempo = 3                ' numero del archivo temporal del listado
CONST numardirec = 4                ' numero del archivo temporal del directorio

CONST maxlpfilas = 58               ' maximo numero de lineas por pagina
CONST maxabonados = 350             ' m ximo numero de abonados por listado telefonico
CONST byblo = 23                    ' bytes por bloque (para equipo)

CONST filinicial = 4                ' fila donde inicia el browse
CONST filinicialar = 8              ' fila inicio lista de archivos
CONST totalfilas = 19               ' total filas del browse de abonados
CONST totalfilasar = 13             ' total filas del browse de archivos
CONST colabonado = 7                ' columna donde posicionar el abonado
CONST colarchivo = 20               ' columna donde posicionar el archivo
CONST colmp1 = 2                    ' columna para popmenu 1: Archivo
```

```
CONST colmp2 = 14                   ' columna para popmenu 2: Comunicacion
CONST colmp3 = 31                   ' columna para popmenu 3: Edicion
CONST colmp4 = 65                   ' columna para popmenu 4: Ayuda

CONST linmenayu = 38                ' longitud de una linea del mensaje de ayuda
CONST filayu = 16                   ' fila para el recuadro de ayuda

CONST filmp = 3                     ' fila para inicio de popmenu
CONST filabr0 = 4                   ' fila inicio lista abonados especiales

CONST nulo = 0                      ' limpia el interior del cuadro
CONST simple = 1                    ' cuadro simple
CONST doble = 2                     ' cuadro doble

CONST cen1 = 101                    ' mensaje centrado en pantalla
CONST cen2 = 102                    ' mensaje centrado a la derecha del listado
CONST col1 = 10                     ' mensaje a partir de col1
CONST col2 = 54                     ' mensaje a partir de col2
CONST col3 = 35                     ' mensaje a partir de col3
CONST colabout = 8                  ' columna para Help/General

CONST alar = 1                      ' rotulo es mensaje de alarma
CONST subt = 2                      ' rotulo es subtitulo
CONST titu = 3                      ' rotulo es titulo
CONST submenuabr = 4                ' rotulo submenu abrir archivo
CONST rotbrowsearchi = 5            ' rotulo subrutina browse archivos para apertura
CONST rotnomarchirx = 6             ' rotulo con nombre de archivo recibido
CONST alarnopcion = 7               ' rotulo es mensaje de alarma sin opcion
CONST submenurx = 8                 ' rotulo submenu recibir listado
CONST mencomunica = 9               ' rotulo subrutinas de comunicacion

CONST si = -1                       ' .T.
CONST no = 0                        ' .F.

CONST careqrx = "Ü"                 ' 219d, 0DBH: caracter valido para recepcion en equipo
CONST careqtx = "½"                 ' 189d, 0BDH: caracter valido para transmision en equipo
CONST carmset = "ì"                 ' 204d, 0CCH: caracter de memoria seteado
CONST carfindir = "^"               '  94d,  5EH: caracter de fin de directorio (listado
telefonico)
```

```
'         ****************
'         MODULO PRINCIPAL
'         ****************

TYPE registro
     NOMBRE    AS STRING * 16
     telefono  AS STRING * 14
     tipo      AS INTEGER
END TYPE
DIM listado AS registro

ON ERROR GOTO errores

inicio:
COLOR 11, 1
CLS
CALL cuadro(doble, 1, 1, 25, 80)
COLOR 14, 1
CALL escrimen(cen1, 3, "ADT96")
CALL escrimen(cen1, 9, "ADITEL")
CALL escrimen(cen1, 10, "ADMINISTRADOR DE DIRECTORIOS TELEFONICOS")
COLOR 7, 1
CALL escrimen(cen1, 17, "team Ronda")
CALL escrimen(cen1, 18, "(c) 1993 - 1996")
CALL escrimen(cen1, 23, "< ▒ >: presione cualquier tecla para continuar")
CALL leerteclado(tecla$)

menuinici% = si
finmp% = no
DO WHILE finmp% = no

menu:
     CALL menu(nomarchivo$, menuinici%, opcion$)

     SELECT CASE opcion$
          CASE IS = "A"
               IF menuinici% = no THEN
                    CALL cerrar(nomarchivo$)
               END IF

               CALL abrir(nomarchivo$)
               IF nomarchivo$ = "*" THEN
                    menuinici% = si
               ELSE
                    menuinici% = no
               END IF

          CASE IS = "B"
               CALL borrar(nomarchivo$)

          CASE IS = "C"
               CALL escrirot(titu, nomarchivo$, opcion%)
               CALL escrirot(subt, "CONSULTAR", opcion%)

               IF LOF(numarchivo) = 0 THEN
                    CALL direstado
               ELSE
                    CALL escrimen(cen2, 14, "< ESC > o < ENTER >: salir")
                    CALL consultar(recno%)
               END IF

          CASE IS = "D"
               CALL definir(nomarchivo$)

          CASE IS = "G"
               CALL about

          CASE IS = "I"
               CALL imprimir(nomarchivo$)

          CASE IS = "M"
               CALL manual
```

```
          CASE IS = "S"
               finmp% = si


          CASE IS = "T"
               CALL transmitir(nomarchivo$)

     END SELECT
LOOP

fin:
IF menuinici% = no THEN
     CALL cerrar(nomarchivo$)
END IF
CLOSE

COLOR 7, 0
CLS
SHELL "del ___(adt).tmp"
SHELL "cd\"
END
```

```
'    ============================
intpuertoserial:
COM(1) OFF
sienlace% = si
GOTO saltorx



'    ============================
errores:
SOUND 2987, 2
COLOR 29, 1
CALL escrimen(cen1, 17, "ERROR")
COLOR 7, 1

SELECT CASE ERR
     CASE IS = 25
          CALL escrimen(cen1, 18, "Impresora apagada o no conectada al computador")
          CALL escrimen(cen1, 19, "< C >: cancelar impresion, < ▓ >: continuar")

     CASE IS = 27
          CALL escrimen(cen1, 18, "Suministro de papel agotado")
          CALL escrimen(cen1, 19, "< C >: cancelar impresion, < ▓ >: continuar")

     CASE IS = 53
          CLS
          SHELL "c:"
          SOUND 2987, 2
          CALL escrimen(cen1, 18, "DISCO PROTEGIDO CONTRA ESCRITURA")
          CALL escrimen(cen1, 19, "< S >: salir del programa, < ▓ >: iniciar nuevamente")

     CASE IS = 71
          CLS
          SHELL "c:"
          SOUND 2987, 2
          CALL escrimen(cen1, 18, "DRIVE NO LISTO")
          CALL escrimen(cen1, 19, "< S >: salir del programa, < ▓ >: iniciar nuevamente")

     CASE ELSE
          CALL escrimen(cen1, 18, "FATAL ERROR - " + LTRIM$(STR$(ERR)))
          CALL escrimen(cen1, 19, "< S >: salir del programa, < ▓ >: iniciar nuevamente")

END SELECT

CALL leerteclado(tecla$)
IF (ERR = 25) OR (ERR = 27) THEN
     IF tecla$ = "C" THEN
          RESUME menu
     ELSE
          CALL cuadro(nulo, 17, 2, 19, 79)
          RESUME
     END IF
ELSE
     IF tecla$ = "S" THEN
          RESUME fin
     ELSE
          IF menuinici% = no THEN
               CALL cerrar(nomarchivo$)
          END IF
          RESUME inicio
     END IF
END IF
```

```
'    ================================================
'    abrir
'    abrir/crear archivo de trabajo, cambiar directorio
'    ================================================
SUB abrir (nomarchivo$)

CLOSE
DIM listarchivos$(1 TO 100)

'    ------------------------------------------------
'    directorio actual y listado de *.rnd que contiene
'    ------------------------------------------------
CLS
CALL direcyarchi(nomdirectorio$, totarchivos%, listarchivos$())
totarchivos$ = LTRIM$(STR$(totarchivos%))

CALL escrirot(titu, "ABRIR / CREAR", opcion%)
COLOR 11, 1
CALL escrimen(cen1, filinicialar - 5, nomdirectorio$)
COLOR 7, 1

archinum% = 1
filactiva% = 1
CALL browsearchi((archinum%), filactiva%, totarchivos%, listarchivos$())
CALL cuadro(simple, filinicialar - 1, colarchivo - 4, filinicialar + totalfilasar, colarchivo +
11)

CALL escrirot(submenuabr, mensaje$, opcion%)
IF totarchivos% > 0 THEN
     CALL escrimen(col3, filinicialar + 3, " < ENTER > : abrir archivo apuntado")
     IF totarchivos% > 1 THEN
          CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
     END IF
END IF

'    --------
'    pantalla
'    --------
finabr% = no
DO WHILE finabr% = no
     IF totarchivos% > 0 THEN
          COLOR 15, 1
          CALL escrimen(colarchivo - 2, filinicialar - 2, CHR$(26) + " " +
LCASE$(listarchivos$(archinum%)))
          COLOR 7, 1
          CALL escrimen(colarchivo, filinicialar + 14, LTRIM$(STR$(archinum%)) + " / " +
totarchivos$ + "   ")
     END IF

     CALL leerteclado(opcion$)

     SELECT CASE opcion$

'    *****     consultar listardo de archivos
          CASE CHR$(0) + CHR$(71), CHR$(0) + CHR$(72), CHR$(0) + CHR$(73), CHR$(0) + CHR$(79),
CHR$(0) + CHR$(80), CHR$(0) + CHR$(81)
               CALL consultarchivo(opcion$, archinum%, filactiva%, totarchivos%, listarchivos$())

'    *****     cambiar de directorio
          CASE IS = "C"
               CALL cuadro(nulo, filinicialar, col3, 20, 79)
               CALL escrimen(colarchivo - 2, filinicialar - 2, STRING$(10, 32))
               CALL escrimen(col3, filinicialar, "     < ESC > : salir")

               CALL ingnuevodir(nuevodirectorio$, escape%)
               IF escape% = si THEN
                    CALL escrimen(col1, filinicialar - 4, STRING$(68, 32))
                    CALL escrirot(submenuabr, mensaje$, opcion%)
                    IF totarchivos% > 0 THEN
                         CALL escrimen(col3, filinicialar + 3, " < ENTER > : abrir archivo
apuntado")
                         IF totarchivos% > 1 THEN
                              CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
                         END IF
```

```
                END IF
            ELSE
                ordendrive$ = LEFT$(nuevodirectorio$, 2)
                nompath$ = MID$(nuevodirectorio$, 3)
                ordenpath$ = "cd " + nompath$

                CLS
                SHELL "del ___(adt).tmp"
                SHELL ordendrive$
                SHELL ordenpath$
                CLS

                CALL direcyarchi(nomdirectorio$, totarchivos%, listarchivos$())
                IF (MID$(nuevodirectorio$, 4) <> MID$(nomdirectorio$, 4)) THEN
                    SOUND 2987, 2
                    COLOR 12, 7
                    CALL cuadro(nulo, 3, 1, 5, 80)
                    CALL escrimen(cen1, 3, CHR$(34) + nuevodirectorio$ + CHR$(34))
                    CALL escrimen(cen1, 4, "DIRECTORIO INDICADO INEXISTENTE")
                    CALL escrimen(cen1, 5, "CAMBIO NO EFECTUADO")
                    COLOR 7, 1
                    CALL escrimen(cen1, 7, "< ▒ >: continuar")
                    CALL leerteclado(tecla$)
                END IF

                totarchivos$ = LTRIM$(STR$(totarchivos%))
                CALL escrirot(titu, "ABRIR / CREAR", opcion%)
                COLOR 11, 1
                CALL escrimen(cen1, filinicialar - 5, nomdirectorio$)
                COLOR 7, 1

                archinum% = 1
                filactiva% = 1
                CALL browsearchi((archinum%), filactiva%, totarchivos%, listarchivos$())
                CALL cuadro(simple, filinicialar - 1, colarchivo - 4, filinicialar +
totalfilasar, colarchivo + 11)

                CALL escrirot(submenuabr, mensaje$, opcion%)
                IF totarchivos% > 0 THEN
                    CALL escrimen(col3, filinicialar + 3, " < ENTER > : abrir archivo
apuntado")
                    IF totarchivos% > 1 THEN
                        CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
                    END IF
                END IF
            END IF

'    *****     archivo nuevo
          CASE IS = "N"
            CALL cuadro(nulo, filinicialar, col3, 20, 79)
            CALL escrimen(colarchivo - 2, filinicialar - 2, STRING$(10, 32))
            CALL escrimen(col3, filinicialar + 4, "     < ESC > : salir")

            CALL escrimen(col3, filinicialar, "ingresar nombre del archivo nuevo")
            CALL ingcadena(nomarchivo$, tiparch, escape%)

            IF escape% = si THEN
                CALL escrirot(submenuabr, mensaje$, opcion%)
                IF totarchivos% > 0 THEN
                    CALL escrimen(col3, filinicialar + 3, " < ENTER > : abrir archivo
apuntado")
                    IF totarchivos% > 1 THEN
                        CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
                    END IF
                END IF
            ELSE
                CALL escrimen(col3, filinicialar + 3, "¨ingreso correcto?")
                CALL escrimen(col3, filinicialar + 4, " < S > : si            ")
                CALL escrimen(col3, filinicialar + 5, " < ▒ > : no")

                CALL leerteclado(tecla$)
                IF tecla$ <> "S" THEN
                    CALL cuadro(nulo, filinicialar + 3, col3, filinicialar + 6, 79)
                    CALL escrirot(submenuabr, mensaje$, opcion%)
```

```
                    IF totarchivos% > 0 THEN
                         CALL escrimen(col3, filinicialar + 3, " < ENTER > : abrir archivo
apuntado")
                         IF totarchivos% > 1 THEN
                              CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
                         END IF
                    END IF
               ELSE
                    yaexiste% = no
                    FOR contador% = 1 TO totarchivos%
                         IF nomarchivo$ = RTRIM$(listarchivos$(contador%)) THEN
                              yaexiste% = si
                              EXIT FOR
                         END IF
                    NEXT contador%

                    IF yaexiste% = no THEN
                         finabr% = si
                         nomarchivo$ = nomarchivo$ + ".rnd"
                         OPEN nomarchivo$ FOR RANDOM AS #numarchivo LEN = longarchivo
                    ELSE
                         SOUND 2987, 2
                         IF totarchivos% <= totalfilasar THEN
                              archinum% = 1
                              filactiva% = contador%
                              CALL browsearchi(archinum%, filactiva%, totarchivos%,
listarchivos$())
                              archinum% = contador%
                         ELSE
                              IF contador% <= totalfilasar THEN
                                   archinum% = 1
                                   filactiva% = contador%
                                   CALL browsearchi(archinum%, filactiva%, totarchivos%,
listarchivos$())
                                   archinum% = contador%
                              ELSE
                                   archinum% = contador% - totalfilasar + 1
                                   filactiva% = totalfilasar
                                   CALL browsearchi(archinum%, filactiva%, totarchivos%,
listarchivos$())
                                   archinum% = archinum% - 1
                              END IF
                         END IF
                         CALL escrimen(colarchivo, filinicialar + 14,
LTRIM$(STR$(archinum%)) + " / " + totarchivos$ + "   ")

                         CALL escrimen(col3, filinicialar + 3, "el archivo ya existe")
                         CALL escrimen(col3, filinicialar + 4, " < A > : abrir")
                         CALL escrimen(col3, filinicialar + 5, " < ▒ > : invalidar ingreso")

                         CALL leerteclado(tecla$)
                         IF tecla$ = "A" THEN
                              finabr% = si
                              nomarchivo$ = nomarchivo$ + ".rnd"
                              OPEN nomarchivo$ FOR RANDOM AS #numarchivo LEN = longarchivo
                         ELSE
                              CALL cuadro(nulo, filinicialar + 5, col3, filinicialar + 6,
79)
                              CALL escrirot(submenuabr, mensaje$, opcion%)
                              IF totarchivos% > 0 THEN
                                   CALL escrimen(col3, filinicialar + 3, " < ENTER > : abrir
archivo apuntado")
                                   IF totarchivos% > 1 THEN
                                        CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
                                   END IF
                              END IF
                         END IF
                    END IF
               END IF
          END IF
          CALL escrimen(col3, filinicialar + 2, STRING$(11, 32))
```

```
'    *****     enter
          CASE IS = CHR$(13)
               IF totarchivos% = 0 THEN
                    SOUND 2987, 2
               ELSE
                    nomarchivo$ = LTRIM$(RTRIM$(listarchivos$(archinum%))) + ".rnd"
                    OPEN nomarchivo$ FOR RANDOM AS #numarchivo LEN = longarchivo
                    finabr% = si
               END IF

'    *****     escape
          CASE IS = CHR$(27)
               nomarchivo$ = "*"
               finabr% = si

          CASE ELSE
               SOUND 2987, 2

     END SELECT
LOOP

END SUB
```

```
'     ======================================
'     borrar
'     borrar abonados del listado telefonico
'     ======================================
SUB borrar (nomarchivo$)

CALL escrirot(titu, nomarchivo$, opcion%)
CALL escrirot(subt, "BORRAR", opcion%)

IF LOF(numarchivo) = 0 THEN
     CALL direstado
     EXIT SUB
END IF

DIM temporal AS registro

finbor% = no
DO WHILE finbor% = no
     CALL cuadro(nulo, 6, 51, 14, 79)
     CALL escrimen(cen2, 13, "< ENTER >: seleccionar")
     CALL escrimen(cen2, 14, " < ESC > : salir      ")
     CALL consultar(recno%)

     IF recno% = 0 THEN
          EXIT DO
     END IF

     CALL cuadro(nulo, 13, 51, 24, 79)

     GET #numarchivo, recno%, temporal
     CALL escrimen(cen2, 6, "¨ borrar abonado " + LTRIM$(STR$(recno%)) + " ?")
     CALL escrimen(cen2, 8, temporal.NOMBRE)
     CALL escrimen(cen2, 9, temporal.telefono + "  ")
     CALL escrimen(cen2, 12, " < B > : borrar            ")
     CALL escrimen(cen2, 13, " < ▒ > : seleccionar otro  ")
     CALL escrimen(cen2, 14, "< ESC >: salir             ")

     CALL leerteclado(tecla$)
     SELECT CASE tecla$
          CASE IS = CHR$(27)
               finbor% = si

          CASE IS = "B"
               numabonados% = LOF(numarchivo) / longarchivo
               IF numabonados% = 1 THEN
                    CLOSE #numarchivo
                    KILL nomarchivo$
                    OPEN nomarchivo$ FOR RANDOM AS #numarchivo LEN = longarchivo

                    CALL escrirot(titu, nomarchivo$, opcion%)
                    CALL escrirot(subt, "BORRAR", opcion%)
                    CALL direstado
                    finbor% = si
               ELSE
                    OPEN "temporal.rnd" FOR RANDOM AS #numartempo LEN = longarchivo

                    FOR reg% = 1 TO recno% - 1
                         GET #numarchivo%, reg%, temporal
                         PUT #numartempo%, reg%, temporal
                    NEXT reg%

                    FOR reg% = recno% + 1 TO LOF(numarchivo%) / longarchivo
                         GET #numarchivo%, reg%, temporal
                         PUT #numartempo%, reg% - 1, temporal
                    NEXT reg%

                    CLOSE #numarchivo
                    KILL nomarchivo$

                    CLOSE #numartempo
                    NAME "temporal.rnd" AS nomarchivo$
                    OPEN nomarchivo$ FOR RANDOM AS #numarchivo LEN = longarchivo
```

```
                    CALL cuadro(nulo, 6, 51, 14, 79)
                    SOUND 2987, 2
                    CALL escrimen(cen2, 11, "-- BORRADO !!")
                    CALL escrimen(cen2, 13, " < ▓ > : seleccionar otro")
                    CALL escrimen(cen2, 14, "< ESC >: salir           ")
                    CALL leerteclado(tecla$)
                    IF tecla$ = CHR$(27) THEN
                         finbor% = si
                    END IF
               END IF

     END SELECT
LOOP

END SUB
```

```
'     ==========================
'     browse
'     mostrar listado telefonico
'     ==========================
SUB browse (recno%, filactiva%, numabonados%)

DIM temporal AS registro

FOR I = 1 TO totalfilas
     fila% = I + filinicial
     LOCATE fila%, colabonado
     IF recno% <= numabonados% THEN
          GET #numarchivo, recno%, temporal

          SELECT CASE temporal.tipo
               CASE 0 TO 9
                    tipo$ = "abr" + LTRIM$(STR$(temporal.tipo))

               CASE 11, 12
                    tipo$ = "hot" + LTRIM$(STR$(temporal.tipo - 10))

               CASE ELSE
                    tipo$ = "    "
          END SELECT

          PRINT STRING$((3 - LEN(LTRIM$(STR$(recno%)))), 32); LTRIM$(STR$(recno%)); "  ";
          PRINT temporal.NOMBRE; " "; temporal.telefono; " "; tipo$;
     ELSE
          PRINT STRING$(40, 32);
     END IF
     recno% = recno% + 1
NEXT
CALL flechabr(filactiva%)

END SUB


'     ==========================
'     browsearchi
'     mostrar listado de archivos
'     ==========================
SUB browsearchi (archinum%, filactiva%, totarchivos%, listarchivos$())

FOR I = 1 TO totalfilasar
     fila% = I + filinicialar - 1
     IF archinum% <= totarchivos% THEN
          CALL escrimen(colarchivo, fila%, listarchivos$(archinum%))
     ELSE
          CALL escrimen(colarchivo, fila%, STRING$(8, 32))
     END IF
     archinum% = archinum% + 1
NEXT
CALL flecharchi(filactiva%, totarchivos%)

END SUB


'     ========================
'     cerrar
'     cerrar archivo de trabajo
'     ========================
SUB cerrar (nomarchivo$)

IF LOF(numarchivo) = 0 THEN
     CLOSE #numarchivo
     KILL nomarchivo$
ELSE
     CLOSE #numarchivo
END IF

END SUB
```

```
'    ============================
'    consultar
'    consultar listado telefonico
'    ============================
SUB consultar (recno%)

CALL cuadro(simple, filinicial - 1, colabonado - 3, filinicial + totalfilas + 1, colabonado + 43)
COLOR 15, 1
LOCATE filinicial, colabonado + 5:  PRINT "nombre";
LOCATE filinicial, colabonado + 22: PRINT "telefono";
LOCATE filinicial, colabonado + 37: PRINT "tipo";

COLOR 7, 1
CALL escrimen(col2, 17, "arriba: < ▓ >")
CALL escrimen(col2, 18, "        < Pg Up >")
CALL escrimen(col2, 19, "        < Home >")
CALL escrimen(col2, 21, " abajo: < ▓ >")
CALL escrimen(col2, 22, "        < Pg Down >")
CALL escrimen(col2, 23, "        < End >")

recno% = 1
filactiva% = 1
numabonados% = LOF(numarchivo) / longarchivo
numabonados$ = LTRIM$(STR$(numabonados%))
CALL browse((recno%), filactiva%, numabonados%)

fincon% = no
DO WHILE fincon% = no
     COLOR 15, 1
     CALL escrimen(cen2, 4, "   " + LTRIM$(STR$(recno%)) + " / " + numabonados$ + "   ")
     COLOR 7, 1

     CALL leerteclado(opcion$)
     SELECT CASE opcion$

'    *****     home: ir al inicio del directorio
          CASE IS = CHR$(0) + CHR$(71)
               recno% = 1
               filactiva% = 1
               CALL browse((recno%), filactiva%, numabonados%)

'    *****     up: ir un abonado arriba
          CASE IS = CHR$(0) + CHR$(72)
               IF filactiva% = 1 THEN
                    IF recno% > 1 THEN
                         recno% = recno% - 1
                         CALL browse((recno%), filactiva%, numabonados%)
                    END IF
               ELSE
                    recno% = recno% - 1
                    filactiva% = filactiva% - 1
                    CALL flechabr(filactiva%)
               END IF

'    *****     page up: ir una p gina arriba
          CASE IS = CHR$(0) + CHR$(73)
               IF filactiva% = 1 THEN
                    IF recno% > totalfilas THEN
                         recno% = recno% - totalfilas
                    ELSE
                         recno% = 1
                    END IF
                    CALL browse((recno%), filactiva%, numabonados%)
               ELSE
                    recno% = recno% - filactiva% + 1
                    filactiva% = 1
                    CALL flechabr(filactiva%)
               END IF

'    *****     end: ir al final del directorio
          CASE IS = CHR$(0) + CHR$(79)
               IF numabonados% <= totalfilas THEN
                    filactiva% = numabonados%
                    CALL flechabr(filactiva%)
```

```
                ELSE
                     recno% = numabonados% - totalfilas + 1
                     filactiva% = totalfilas
                     CALL browse(recno%, filactiva%, numabonados%)
                END IF
                recno% = numabonados%

'    *****     down: ir un abonado abajo
          CASE IS = CHR$(0) + CHR$(80)
                IF numabonados% <= totalfilas THEN
                     IF filactiva% < numabonados% THEN
                          recno% = recno% + 1
                          filactiva% = filactiva% + 1
                          CALL flechabr(filactiva%)
                     END IF
                ELSE
                     IF filactiva% < totalfilas THEN
                          recno% = recno% + 1
                          filactiva% = filactiva% + 1
                          CALL flechabr(filactiva%)
                     ELSE
                          IF recno% < numabonados% THEN
                               recno% = recno% - totalfilas + 2
                               CALL browse(recno%, filactiva%, numabonados%)
                               recno% = recno% - 1
                          END IF
                     END IF
                END IF

'    *****     page down: ir una p gina abajo
          CASE IS = CHR$(0) + CHR$(81)
                IF numabonados% <= totalfilas THEN
                     recno% = numabonados%
                     filactiva% = numabonados%
                     CALL flechabr(filactiva%)
                ELSE
                     IF filactiva% < totalfilas THEN
                          recno% = recno% + totalfilas - filactiva%
                          filactiva% = totalfilas
                          CALL flechabr(filactiva%)
                     ELSE
                          IF recno% + totalfilas > numabonados% THEN
                               recno% = numabonados% - totalfilas + 1
                          ELSE
                               recno% = recno% + 1
                          END IF
                          CALL browse(recno%, filactiva%, numabonados%)
                          recno% = recno% - 1
                     END IF
                END IF

'    *****     enter
          CASE IS = CHR$(13)
                fincon% = si

'    *****     escape
          CASE IS = CHR$(27)
                recno% = 0
                fincon% = si

          CASE ELSE
                SOUND 2934, 2

     END SELECT
LOOP

END SUB
```

```
'    ================================================
'    consultarchivo
'    consultar el contenido del arreglo listarchivos$()
'    ================================================
SUB consultarchivo (opcion$, archinum%, filactiva%, totarchivos%, listarchivos$())

SELECT CASE opcion$

'    *****     home: ir al inicio del listado
     CASE IS = CHR$(0) + CHR$(71)
          IF totarchivos% < 2 THEN
               SOUND 2987, 2
          ELSE
               archinum% = 1
               filactiva% = 1
               CALL browsearchi((archinum%), filactiva%, totarchivos%, listarchivos$())
          END IF

'    *****     up: ir un abonado arriba
     CASE IS = CHR$(0) + CHR$(72)
          IF totarchivos% < 2 THEN
               SOUND 2987, 2
          ELSE
               IF filactiva% = 1 THEN
                    IF archinum% > 1 THEN
                         archinum% = archinum% - 1
                         CALL browsearchi((archinum%), filactiva%, totarchivos%, listarchivos$())
                    END IF
               ELSE
                    archinum% = archinum% - 1
                    filactiva% = filactiva% - 1
                    CALL flecharchi(filactiva%, totarchivos%)
               END IF
          END IF

'    *****     page up: ir una p gina arriba
     CASE IS = CHR$(0) + CHR$(73)
          IF totarchivos% < 2 THEN
               SOUND 2987, 2
          ELSE
               IF filactiva% = 1 THEN
                    IF archinum% > totalfilas THEN
                         archinum% = archinum% - totalfilasar
                    ELSE
                         archinum% = 1
                    END IF
                    CALL browsearchi((archinum%), filactiva%, totarchivos%, listarchivos$())
               ELSE
                    archinum% = archinum% - filactiva% + 1
                    filactiva% = 1
                    CALL flecharchi(filactiva%, totarchivos%)
               END IF
          END IF


'    *****     end: ir al final del listado
     CASE IS = CHR$(0) + CHR$(79)
          IF totarchivos% < 2 THEN
               SOUND 2987, 2
          ELSE
               IF totarchivos% <= totalfilasar THEN
                    filactiva% = totarchivos%
                    CALL flecharchi(filactiva%, totarchivos%)
               ELSE
                    archinum% = totarchivos% - totalfilasar + 1
                    filactiva% = totalfilasar
                    CALL browsearchi(archinum%, filactiva%, totarchivos%, listarchivos$())
               END IF
               archinum% = totarchivos%
          END IF
```

```
'    *****     down: ir un abonado abajo
     CASE IS = CHR$(0) + CHR$(80)
          IF totarchivos% < 2 THEN
               SOUND 2987, 2
          ELSE
               IF totarchivos% <= totalfilasar THEN
                    IF filactiva% < totarchivos% THEN
                         archinum% = archinum% + 1
                         filactiva% = filactiva% + 1
                         CALL flecharchi(filactiva%, totarchivos%)
                    END IF
               ELSE
                    IF filactiva% < totalfilasar THEN
                         archinum% = archinum% + 1
                         filactiva% = filactiva% + 1
                         CALL flecharchi(filactiva%, totarchivos%)
                    ELSE
                         IF archinum% < totarchivos% THEN
                              archinum% = archinum% - totalfilasar + 2
                              CALL browsearchi(archinum%, filactiva%, totarchivos%,
listarchivos$())
                              archinum% = archinum% - 1
                         END IF
                    END IF
               END IF
          END IF

'    *****     page down: ir una p gina abajo
     CASE IS = CHR$(0) + CHR$(81)
          IF totarchivos% < 2 THEN
               SOUND 2987, 2
          ELSE
               IF totarchivos% <= totalfilasar THEN
                    archinum% = totarchivos%
                    filactiva% = totarchivos%
                    CALL flecharchi(filactiva%, totarchivos%)
               ELSE
                    IF filactiva% < totalfilasar THEN
                         archinum% = archinum% + totalfilasar - filactiva%
                         filactiva% = totalfilasar
                         CALL flecharchi(filactiva%, totarchivos%)
                    ELSE
                         IF archinum% + totalfilasar > totarchivos% THEN
                              archinum% = totarchivos% - totalfilasar + 1
                         ELSE
                              archinum% = archinum% + 1
                         END IF
                         CALL browsearchi(archinum%, filactiva%, totarchivos%, listarchivos$())
                         archinum% = archinum% - 1
                    END IF
               END IF
          END IF

END SELECT

END SUB
```

```
'     ======================================
'     cuadro
'     dibujar un cuadro nulo, simple o doble
'     ======================================
SUB cuadro (tipcuad%, fili%, coli%, filf%, colf%)

SELECT CASE tipcuad%
     CASE IS = nulo
          blancos$ = STRING$(colf% - coli% + 1, 32)
          FOR I = fili% TO filf%
               LOCATE I, coli%
               PRINT blancos$;
          NEXT I

     CASE IS = simple
          horizontal$ = STRING$(colf% - coli% - 1, 196)

          LOCATE fili%, coli%: PRINT CHR$(218); horizontal$;
          LOCATE fili%, colf%: PRINT CHR$(191);
          LOCATE filf%, coli%: PRINT CHR$(192); horizontal$;
          LOCATE filf%, colf%: PRINT CHR$(217);

          FOR I = fili% + 1 TO filf% - 1
               LOCATE I, coli%: PRINT CHR$(179);
               LOCATE I, colf%: PRINT CHR$(179);
          NEXT I

     CASE IS = doble
          horizontal$ = STRING$(colf% - coli% - 1, 205)

          LOCATE fili%, coli%: PRINT CHR$(201); horizontal$;
          LOCATE fili%, colf%: PRINT CHR$(187);
          LOCATE filf%, coli%: PRINT CHR$(200); horizontal$;
          LOCATE filf%, colf%: PRINT CHR$(188);

          FOR I = fili% + 1 TO filf% - 1
               LOCATE I, coli%: PRINT CHR$(186);
               LOCATE I, colf%: PRINT CHR$(186);
          NEXT I

END SELECT

END SUB
```

```
'    ========================
'    definir
'    definir tipo de abonado
'    ========================
SUB definir (nomarchivo$)

IF LOF(numarchivo) = 0 THEN
     CALL direstado
     EXIT SUB
END IF

DIM temporal AS registro
numabonados% = LOF(numarchivo) / longarchivo
numabonados$ = LTRIM$(STR$(numabonados%))

DIM especial%(0 TO 12)
FOR recno% = 1 TO numabonados%                          ' almacenar los abonados especiales
     GET #numarchivo, recno%, temporal
     IF temporal.tipo <> 10 THEN
          especial%(temporal.tipo) = recno%
     END IF
NEXT recno%

findef% = no
DO WHILE findef% = no
     CALL escrirot(titu, nomarchivo$, opcion%)
     CALL escrirot(subt, "DEFINIR ABONADOS ESPECIALES", opcion%)

     CALL escrimen(col1, filabr0 + 0, " < 0 > : abr0")
     CALL escrimen(col1, filabr0 + 1, " < 1 > : abr1")
     CALL escrimen(col1, filabr0 + 2, " < 2 > : abr2")
     CALL escrimen(col1, filabr0 + 3, " < 3 > : abr3")
     CALL escrimen(col1, filabr0 + 4, " < 4 > : abr4")
     CALL escrimen(col1, filabr0 + 5, " < 5 > : abr5")
     CALL escrimen(col1, filabr0 + 6, " < 6 > : abr6")
     CALL escrimen(col1, filabr0 + 7, " < 7 > : abr7")
     CALL escrimen(col1, filabr0 + 8, " < 8 > : abr8")
     CALL escrimen(col1, filabr0 + 9, " < 9 > : abr9")
     CALL escrimen(col1, filabr0 + 11, " < F2 >: hot1")
     CALL escrimen(col1, filabr0 + 12, " < F3 >: hot2")

     CALL escrimen(col1, filabr0 + 14, " < F9 >: todos normales")
     CALL escrimen(col1, filabr0 + 15, "< ESC >: salir")
     CALL escrimen(cen1, 23, "Presione la < tecla > para ejecutar la accion.")

     FOR I = 0 TO 12
          IF especial%(I) <> 0 THEN
               GET #numarchivo, especial%(I), temporal
               especial$ = LTRIM$(STR$(especial%(I)))
               rec$ = STRING$(LEN(numabonados$) - LEN(especial$), 32) + "[" + especial$ + "/" +
numabonados$ + "]"
               LOCATE filabr0 + I, 25: PRINT rec$; "  "; temporal.NOMBRE; "  ";
temporal.telefono;
          END IF
     NEXT I

     teclavalida% = no
     DO WHILE teclavalida% = no
          CALL leerteclado(tecla$)
          IF (tecla$ >= "0" AND tecla$ <= "9") THEN
               teclavalida% = si
          ELSEIF (tecla$ = CHR$(0) + CHR$(60) OR tecla$ = CHR$(0) + CHR$(61) OR tecla$ =
CHR$(27)) THEN
               teclavalida% = si
          ELSEIF tecla$ = CHR$(0) + CHR$(67) THEN
               CALL escrimen(cen1, 23, "                                                    ")
               CALL escrirot(alar, "DEFINIR A TODOS LOS ABONADOS DEL LISTADO COMO NORMALES",
teclavalida%)
               IF teclavalida% = no THEN
                    CALL escrimen(cen1, 23, "Presione la < tecla > para ejecutar la accion.")
               END IF
          ELSE
               SOUND 2987, 2
          END IF
```

```
     LOOP
     SELECT CASE tecla$

'    *****     tipo de abonado especial
          CASE "0" TO "9", CHR$(0) + CHR$(60), CHR$(0) + CHR$(61)
               IF tecla$ = CHR$(0) + CHR$(60) THEN
                    mensaje$ = "HOT PARA LT-1"
                    tipo% = 11
               ELSEIF tecla$ = CHR$(0) + CHR$(61) THEN
                    mensaje$ = "HOT PARA LT-2"
                    tipo% = 12
               ELSE
                    mensaje$ = "ABREVIADO " + tecla$
                    tipo% = VAL(tecla$)
               END IF

               CALL escrirot(titu, nomarchivo$, opcion%)
               CALL escrirot(subt, "DEFINIR ABONADOS ESPECIALES", opcion%)

               CALL escrimen(cen2, 8, "definir")
               COLOR 15, 1
               CALL escrimen(cen2, 9, mensaje$)
               COLOR 7, 1
               CALL escrimen(cen2, 13, "< ENTER >: seleccionar")
               CALL escrimen(cen2, 14, "  < ESC >: salir      ")

               CALL consultar(recno%)
               IF recno% <> 0 THEN
                    IF especial%(tipo%) <> 0 THEN
                         GET #numarchivo, especial%(tipo%), temporal
                         temporal.tipo = 10
                         PUT #numarchivo, especial%(tipo%), temporal
                    END IF

                    GET #numarchivo, recno%, temporal            ' definir al abonado como
especial
                    especial%(temporal.tipo) = 0
                    temporal.tipo = tipo%
                    PUT #numarchivo, recno%, temporal
                    especial%(tipo%) = recno%
               END IF

'    *****     todos normales < F9 >
          CASE IS = CHR$(0) + CHR$(67)
               FOR I = 0 TO 12
                    IF especial%(I) <> 0 THEN
                         GET #numarchivo, especial%(I), temporal
                         temporal.tipo = 10
                         PUT #numarchivo, especial%(I), temporal
                         especial%(I) = 0
                    END IF
               NEXT I

'    *****     escape
          CASE IS = CHR$(27)
               findef% = si

     END SELECT
LOOP

END SUB
```

```
'     ================================
'     direcyarchi
'     directorio y archivo de trabajo
'     ================================
SUB direcyarchi (nomdirectorio$, totarchivos%, listarchivos$())

nomdirectorio$ = "*"

DO WHILE nomdirectorio$ = "*"
     SHELL "dir /a:-d /o:n *.rnd > ___(adt).tmp"
     OPEN "___(adt).tmp" FOR INPUT AS #numardirec

     totarchivos% = 0
     contador% = 1
     DO WHILE NOT EOF(numardirec)
          LINE INPUT #numardirec, linea$

          IF contador% = 4 THEN
               linea$ = LTRIM$(RTRIM$(linea$))
               nomdirectorio$ = MID$(linea$, 14)
          ELSEIF (contador% > 5 AND LEFT$(linea$, 1) <> CHR$(32)) THEN
               totarchivos% = totarchivos% + 1
               linea$ = LEFT$(LTRIM$(RTRIM$(linea$)), 8)
               listarchivos$(totarchivos%) = linea$
          END IF
          contador% = contador% + 1
     LOOP
     CLOSE numardirec
LOOP

END SUB



'     ============================================================
'     direstado
'     indica si el directorio de trabajo se halla vacio o lleno
'     ============================================================
SUB direstado

IF LOF(numarchivo) = 0 THEN
     mensaje$ = "DIRECTORIO VACIO"
ELSE
     mensaje$ = "DIRECTORIO LLENO"
END IF
CALL escrirot(alarnopcion, mensaje$, 18)

END SUB
```

```
'     ================
'     escrimen
'     escribir mensaje
'     ================
SUB escrimen (tipmen%, fila%, mensaje$)

SELECT CASE tipmen%
     CASE IS = cen1
          LOCATE fila%, INT((80 - LEN(mensaje$)) / 2)

     CASE IS = cen2
          LOCATE fila%, 51 + INT((30 - LEN(mensaje$)) / 2)

     CASE ELSE
          LOCATE fila%, tipmen%

END SELECT

FOR I = 1 TO LEN(mensaje$)
     IF MID$(mensaje$, I, 1) = "<" THEN
          COLOR 7, 1
          PRINT MID$(mensaje$, I, 1);
          COLOR 15, 1
     ELSEIF MID$(mensaje$, I, 1) = ">" THEN
          COLOR 7, 1
          PRINT MID$(mensaje$, I, 1);
     ELSE
          PRINT MID$(mensaje$, I, 1);
     END IF
NEXT I

END SUB
```

```
'    ===============
'    escrirot
'    escribir rotulo
'    ===============
SUB escrirot (tiprot%, mensaje$, opcion%)

SELECT CASE tiprot%
     CASE IS = alar
          SOUND 2987, 2
          COLOR 13, 1
          CALL cuadro(simple, 21, (32 - INT(LEN(mensaje$) / 2)), 24, (47 + INT(LEN(mensaje$) /
2)))
          COLOR 11, 1
          mensaje$ = "¨ " + mensaje$ + " ?"
          CALL escrimen(cen1, 22, mensaje$)
          CALL escrimen(cen1, 23, "< S >: si      < ▒ >: no")
          COLOR 7, 1

          CALL leerteclado(tecla$)
          IF tecla$ = "S" THEN
               opcion% = si
          ELSE
               opcion% = no
          END IF
          CALL cuadro(nulo, 21, (32 - INT(LEN(mensaje$) / 2)), 24, (47 + INT(LEN(mensaje$) / 2)))

     CASE IS = subt
          COLOR 11, 1
          CALL escrimen(cen1, 2, mensaje$)
          COLOR 7, 1

     CASE IS = titu
          CLS
          COLOR 14, 1
          CALL cuadro(doble, 1, 1, 25, 80)
          CALL escrimen(cen1, 1, "  ADITEL-pc96 : " + mensaje$ + "  ")
          CALL escrimen(cen1, 25, "  team Ronda  ")
          COLOR 7, 1

     CASE IS = submenuabr
          CALL escrimen(col3, filinicialar + 0, "    < C >  : cambiar directorio")
          CALL escrimen(col3, filinicialar + 1, "    < N >  : archivo nuevo")
          CALL escrimen(col3, filinicialar + 4, "   < ESC > : salir  ")

     CASE IS = rotbrowsearchi
          CALL escrimen(col3, 18, "arriba: < ▒ >       abajo: < ▒ >")
          CALL escrimen(col3, 19, "        < Pg Up >          < Pg Down >")
          CALL escrimen(col3, 20, "        < Home >           < End >")

     CASE IS = rotnomarchirx
          CALL escrimen(colarchivo - 9, filinicialar - 2, "nombre del listado recibido: ")
          COLOR 15, 1
          IF LEN(mensaje$) = 0 THEN
               CALL escrimen(colarchivo + 20, filinicialar - 2, "SIN NOMBRE")
          ELSE
               CALL escrimen(colarchivo + 20, filinicialar - 2, mensaje$)
          END IF
          COLOR 7, 1

     CASE IS = alarnopcion
          SOUND 2987, 2
          COLOR 11, 1
          CALL cuadro(simple, 21, (32 - INT(LEN(mensaje$) / 2)), 24, (47 + INT(LEN(mensaje$) /
2)))
          CALL escrimen(cen1, 22, mensaje$)
          CALL escrimen(cen1, 23, "< ▒ >: continuar")
          CALL leerteclado(tecla$)
          CALL cuadro(nulo, 21, (32 - INT(LEN(mensaje$) / 2)), 24, (47 + INT(LEN(mensaje$) / 2)))

     CASE IS = submenurx
          CALL escrimen(col3, filinicialar + 0, " < C >  : cambiar directorio         ")
          CALL escrimen(col3, filinicialar + 1, " < N >  : cambiar nombre del listado ")
          CALL escrimen(col3, filinicialar + 2, " < G >  : grabar listado recibido   ")
          CALL escrimen(col3, filinicialar + 4, "< ESC > : salir      ")
```

```
      CASE IS = mencomunica
           CALL escrimen(cen1, 6, "REVISAR QUE:")
           CALL escrimen(cen1, 8, "*  en la pantalla del equipo aparezca       ")
           CALL escrimen(cen1, 9, "   el mensaje " + CHR$(34) + "MODO SERIAL" + CHR$(34) + ",
")
           CALL escrimen(cen1, 10, "*  el cable de comunicaciones se halle      ")
           CALL escrimen(cen1, 11, "   adecuadamente conectado al computador, y,")
           CALL escrimen(cen1, 12, "*  la luz SERIAL del equipo (primera de     ")
           CALL escrimen(cen1, 13, "   la derecha) este encendida.              ")

END SELECT

END SUB
```

```
'    ======================================
'    flechabr
'    posicionar flecha en listado telefonico
'    ======================================
SUB flechabr (filactiva%)

COLOR 15, 1
FOR I = 1 TO totalfilas
     IF I = filactiva% THEN
          car1$ = CHR$(16)
          car2$ = CHR$(17)
     ELSE
          car1$ = CHR$(32)
          car2$ = CHR$(32)
     END IF

     fila% = I + filinicial
     LOCATE fila%, colabonado - 2: PRINT car1$;
     LOCATE fila%, colabonado + 42: PRINT car2$;
NEXT
COLOR 7, 1

END SUB



'    ======================================
'    flecharchi
'    posicionar flecha en listado de archivos
'    ======================================
SUB flecharchi (filactiva%, totarchivos%)

IF totarchivos% = 0 THEN
     EXIT SUB
END IF

COLOR 15, 1
FOR I = 1 TO totalfilasar
     IF I = filactiva% THEN
          car1$ = CHR$(16)
          car2$ = CHR$(17)
     ELSE
          car1$ = CHR$(32)
          car2$ = CHR$(32)
     END IF

     fila% = I + filinicialar - 1
     LOCATE fila%, colarchivo - 2: PRINT car1$;
     LOCATE fila%, colarchivo + 9: PRINT car2$;
NEXT
COLOR 7, 1

END SUB
```

```
'    ============================
'    flechamp
'    posicionar flecha en popmenu
'    ============================
SUB flechamp (numenu%, nupop%)

SELECT CASE numenu%

'    *****     menu ARCHIVO (en inicializacion)
     CASE IS = 0
          IF nupop% = 0 THEN
               nupop% = 3
          ELSEIF nupop% = 4 THEN
               nupop% = 1
          END IF

          colflechamp1% = colmp1 + 2
          colflechamp2% = colmp1 + 18
          ultfilmp% = 4
          IF nupop% = 3 THEN
               filflechamp% = nupop% + 3
          ELSE
               filflechamp% = nupop% + 2
          END IF

'    *****     menu ARCHIVO (en principal)
     CASE IS = 1
          IF nupop% = 0 THEN
               nupop% = 4
          ELSEIF nupop% = 5 THEN
               nupop% = 1
          END IF

          colflechamp1% = colmp1 + 2
          colflechamp2% = colmp1 + 18
          ultfilmp% = 6
          IF nupop% = 4 THEN
               filflechamp% = nupop% + 4
          ELSEIF nupop% > 1 THEN
               filflechamp% = nupop% + 3
          ELSE
               filflechamp% = nupop% + 2
          END IF

'    *****     menu COMUNICACION
     CASE IS = 2
          IF nupop% = 0 THEN
               nupop% = 2
          ELSEIF nupop% = 3 THEN
               nupop% = 1
          END IF

          filflechamp% = nupop% + 2
          colflechamp1% = colmp2 + 2
          colflechamp2% = colmp2 + 16
          ultfilmp% = 2

'    *****     menu EDITAR
     CASE IS = 3
          IF nupop% = 0 THEN
               nupop% = 3
          ELSEIF nupop% = 4 THEN
               nupop% = 1
          END IF

          filflechamp% = nupop% + 2
          colflechamp1% = colmp3 + 2
          colflechamp2% = colmp3 + 12
          ultfilmp% = 3

'    *****     menu HELP
     CASE IS = 4
          IF nupop% = 0 THEN
               nupop% = 2
```

```
            ELSEIF nupop% = 3 THEN
                 nupop% = 1
            END IF

            filflechamp% = nupop% + 2
            colflechamp1% = colmp4 + 2
            colflechamp2% = colmp4 + 12
            ultfilmp% = 2

END SELECT

COLOR 15, 1
FOR fila% = filmp TO ultfilmp% + filmp - 1
     IF fila% = filflechamp% THEN
          car1$ = CHR$(16)
          car2$ = CHR$(17)
     ELSE
          car1$ = CHR$(32)
          car2$ = CHR$(32)
     END IF

     LOCATE fila%, colflechamp1%: PRINT car1$;
     LOCATE fila%, colflechamp2%: PRINT car2$;
NEXT fila%

numayuda% = 10 * numenu% + nupop%
CALL menayuda(numayuda%)

END SUB
```

```
'    ============================
'    imprimir
'    imprimir listado telefonico
'    ============================
SUB imprimir (nomarchivo$)

CALL escrirot(titu, nomarchivo$, opcion%)
CALL escrirot(subt, "IMPRIMIR", opcion%)

IF LOF(numarchivo) = 0 THEN
     CALL direstado
     EXIT SUB
END IF

DIM temporal AS registro
numabonados% = LOF(numarchivo) / longarchivo
numabonados$ = LTRIM$(STR$(numabonados%))
mensaje1$ = "MARCADOR TELEFONICO AUTOMATICO"
mensaje2$ = "DIRECTORIO TELEFONICO"
nomgrupo$ = STRING$(17, 32)
grupoant$ = ""
numero$ = ""
numerar% = no
fecha% = si
pagina% = si
calidad% = no
separar% = si

CALL escrimen(15, 6, "[   ]   < A > :   numerar abonados")
CALL escrimen(15, 7, "[   ]   < F > :   imprimir fecha")
CALL escrimen(15, 8, "[   ]   < P > :   numerar p gina")
CALL escrimen(15, 9, "[   ]   < O > :   alta calidad de impresion")
CALL escrimen(15, 10, "[   ]   < S > :   separar abonados por grupos")
CALL escrimen(22, 12, "< F7 > :   imprimir listado telefonico")
CALL escrimen(22, 13, "< ESC >:   salir")
CALL escrimen(cen1, 15, "Seleccionar opcion")

finimp% = no
DO WHILE finimp% = no
     COLOR 14, 1
     LOCATE 6, 17
     IF numerar% = si THEN
          PRINT "X";
     ELSE
          PRINT CHR$(32);
     END IF

     LOCATE 7, 17
     IF fecha% = si THEN
          PRINT "X";
     ELSE
          PRINT CHR$(32);
     END IF

     LOCATE 8, 17
     IF pagina% = si THEN
          PRINT "X";
     ELSE
          PRINT CHR$(32);
     END IF

     LOCATE 9, 17
     IF calidad% = si THEN
          PRINT "X";
     ELSE
          PRINT CHR$(32);
     END IF

     LOCATE 10, 17
     IF separar% = si THEN
          PRINT "X";
     ELSE
          PRINT CHR$(32);
     END IF
```

```
CALL leerteclado(tecla$)
SELECT CASE tecla$
     CASE IS = "A"
          IF numerar% = si THEN
               numerar% = no
          ELSE
               numerar% = si
          END IF

     CASE IS = "F"
          IF fecha% = si THEN
               fecha% = no
          ELSE
               fecha% = si
          END IF

     CASE IS = "P"
          IF pagina% = si THEN
               pagina% = no
          ELSE
               pagina% = si
          END IF

     CASE IS = "Q"
          IF calidad% = si THEN
               calidad% = no
          ELSE
               calidad% = si
          END IF

     CASE IS = "S"
          IF separar% = si THEN
               separar% = no
          ELSE
               separar% = si
          END IF

     CASE IS = CHR$(0) + CHR$(65)
          CALL escrirot(alar, "IMPRIMIR DIRECTORIO " + nomarchivo$, siimp%)

          IF siimp% = si THEN
               CALL escrimen(cen1, 18, "imprimiendo")

               IF calidad% = si THEN
                    LPRINT CHR$(27); "x"; "1"
               ELSE
                    LPRINT CHR$(27); "x"; "0"
               END IF

               IF (fecha% = si) AND (pagina% = si) THEN          ' encabezado pag 1
                    LPRINT STRING$(69, 32); DATE$
                    LPRINT STRING$(69, 32); "pag 1"
               ELSEIF (fecha% = si) AND (pagina% = no) THEN
                    LPRINT STRING$(69, 32); DATE$
                    LPRINT
               ELSEIF (fecha% = no) AND (pagina% = si) THEN
                    LPRINT STRING$(69, 32); "pag 1"
                    LPRINT
               ELSE
                    LPRINT
                    LPRINT
               END IF

               LPRINT
               LPRINT STRING$(40 - INT(LEN(mensaje1$) / 2), 32); mensaje1$
               LPRINT
               LPRINT STRING$(40 - INT(LEN(mensaje2$) / 2), 32); mensaje2$
               LPRINT STRING$(40 - INT(LEN(nomarchivo$) / 2), 32); nomarchivo$
               LPRINT
               LPRINT
               LPRINT STRING$(10, 32); "NUMERO DE ABONADOS: "; numabonados%
               LPRINT
```

```
                    fila% = 13
                    recno% = 1
                    pag% = 1
                    DO WHILE recno% <= numabonados%
                         IF fila% = 4 THEN
                              LPRINT STRING$(69, 32); nomarchivo$              ' encabezado pag>1
                              IF (fecha% = si) AND (pagina% = si) THEN
                                   LPRINT STRING$(69, 32); DATE$
                                   LPRINT STRING$(69, 32); "pag"; STR$(pag%)
                              ELSEIF (fecha% = si) AND (pagina% = no) THEN
                                   LPRINT STRING$(69, 32); DATE$
                                   LPRINT
                              ELSEIF (fecha% = no) AND (pagina% = si) THEN
                                   LPRINT STRING$(69, 32); "pag"; STR$(pag%)
                                   LPRINT
                              ELSE
                                   LPRINT
                                   LPRINT
                              END IF
                              LPRINT
                         END IF

                         CALL escrimen(cen1, 19, LTRIM$(STR$(recno%)) + " / " + numabonados$)
                         GET #numarchivo, recno%, temporal
                         SELECT CASE temporal.tipo
                              CASE 0 TO 9
                                   tipo$ = "abr " + LTRIM$(STR$(temporal.tipo))

                              CASE 11, 12
                                   tipo$ = "hot lt" + LTRIM$(STR$(temporal.tipo - 10))

                              CASE ELSE
                                   tipo$ = ""
                         END SELECT

                         IF separar% = si THEN
                              grupo$ = LEFT$(temporal.NOMBRE, 1)
                              IF grupoant$ <> grupo$ THEN
                                   LPRINT
                                   fila% = fila% + 1
                                   nomgrupo$ = STRING$(12, 32) + grupo$ + "    "
                                   grupoant$ = grupo$
                              ELSE
                                   nomgrupo$ = STRING$(17, 32)
                              END IF
                         END IF

                         IF numerar% = si THEN
                              numero$ = LTRIM$(STR$(recno%))
                              numero$ = STRING$(3 - LEN(numero$), 32) + numero$ + "  "
                         END IF

                         LPRINT nomgrupo$; numero$; temporal.NOMBRE; "  "; temporal.telefono; "
"; tipo$
                         fila% = fila% + 1
                         recno% = recno% + 1
                         IF (recno% > numabonados%) OR (fila% > maxlpfilas) THEN
                              fila% = 4
                              pag% = pag% + 1
                              LPRINT CHR$(12);
                         END IF
                    LOOP

                    numero$ = ""
                    calidad% = no
                    CALL escrimen(cen1, 18, "           ")
                    CALL escrimen(cen1, 19, "          ")
                    SOUND 2987, 2
               END IF

          CASE IS = CHR$(27)
               finimp% = si
```

```
          CASE ELSE
               SOUND 2987, 2
     END SELECT
LOOP
COLOR 7, 1

END SUB
```

```
'    ===============
'    ingcadena
'    ingresar cadena
'    ===============
SUB ingcadena (cadena$, tipo$, escape%)

escape% = no
SELECT CASE tipo$
     CASE IS = tiparch
          cadena$ = STRING$(8, 250)
          colu% = col3 + 5
          fila% = filinicialar + 1

     CASE IS = tipnoma
          cadena$ = STRING$(16, 250)
          mensaje$ = "                  nombre : "
          colu% = 38
          fila% = 10
          CALL cuadro(nulo, 11, 10, 14, 60)                  ' borrar telefono y mensaje final

     CASE IS = tiptela
          cadena$ = STRING$(14, 250)
          mensaje$ = "                telefono : "
          colu% = 38
          fila% = 11

END SELECT

fincadena% = LEN(cadena$) + 1

IF tipo$ <> tiparch THEN
     CALL escrimen(col1, fila%, mensaje$)
     CALL escrimen(colu%, fila% + 2, "< ESC >: salir")
     CALL escrimen(cen1, 22, "Caracteres v lidos:  nombre: A - Z, espacio en blanco")
     CALL escrimen(cen1, 23, "                     numero: 0 - 9                    ")
END IF

COLOR 15, 1
CALL escrimen(colu%, fila%, cadena$)
COLOR 31, 1: LOCATE fila%, colu%: PRINT "_";
COLOR 15, 1

poscar% = 1
fininc% = no                                                 ' fin ingreso cadena
DO WHILE fininc% = no
     CALL leerteclado(tecla$)

     SELECT CASE tecla$
          CASE IS = CHR$(32)                                 ' blanco
               IF (tipo$ = tipnoma) AND (poscar% <> 1) THEN
                    MID$(cadena$, poscar%, 1) = tecla$
                    poscar% = poscar% + 1
               ELSE
                    SOUND 2987, 2
               END IF

          CASE "0" TO "9"                                    ' 0 - 9
               IF (tipo$ = tipnoma) THEN
                    SOUND 2987, 2
               ELSE
                    MID$(cadena$, poscar%, 1) = tecla$
                    poscar% = poscar% + 1
               END IF

          CASE "A" TO "Z"                                    ' A - Z
               IF (tipo$ = tiptela) THEN
                    SOUND 2987, 2
               ELSE
                    MID$(cadena$, poscar%, 1) = tecla$
                    poscar% = poscar% + 1
               END IF
```

```
          CASE IS = CHR$(8)                                        ' backspace
               IF poscar% = 1 THEN
                    SOUND 2987, 2                                  ' cadena vacia
               ELSE
                    poscar% = poscar% - 1
                    MID$(cadena$, poscar%, 1) = CHR$(250)
               END IF

          CASE IS = CHR$(13)                                       ' enter
               IF poscar% = 1 THEN
                    SOUND 2987, 2                                  ' cadena vacia
               ELSE
                    fininc% = si
               END IF

          CASE IS = CHR$(27)                                       ' escape
               escape% = si
               fininc% = si

          CASE ELSE
               SOUND 2987, 2
     END SELECT

     CALL escrimen(colu%, fila%, cadena$)
     IF poscar% = fincadena% THEN
          fininc% = si
          SOUND 2987, 2
     ELSE
          COLOR 31, 1: LOCATE fila%, colu% + poscar% - 1: PRINT "_";
          COLOR 15, 1
     END IF
LOOP

CALL escrimen(colu% + poscar% - 1, fila%, STRING$(fincadena% - poscar%, " "))

FOR I = 1 TO fincadena%                                            ' quitar los CHR$(250) del
     IF MID$(cadena$, I, 1) = CHR$(250) THEN                       ' final de la cadena
          cadena$ = LEFT$(cadena$, I - 1)
          EXIT FOR
     END IF
NEXT I

COLOR 7, 1

IF tipo$ <> tiparch THEN
     CALL escrimen(colu%, fila% + 2, "              ")
END IF

END SUB
```

```
'    ========================================
'    ingnuevodir
'    ingresar nombre para el nuevo directorio
'    ========================================
SUB ingnuevodir (nuevodirectorio$, escape%)

nuevodirectorio$ = STRING$(40, 250)
CALL escrimen(col1, filinicialar - 4, "Nuevo directorio:")
COLOR 15, 1: LOCATE filinicialar - 4, 30: PRINT nuevodirectorio$;
COLOR 31, 1: LOCATE filinicialar - 4, 30: PRINT "_";
COLOR 15, 1

fincadena% = LEN(nuevodirectorio$) + 1
poscar% = 1
escape% = no
finuevodir% = no
DO WHILE finuevodir% = no
     CALL leerteclado(tecla$)

     SELECT CASE tecla$
          CASE "A" TO "H"
               IF poscar% = 1 THEN
                    MID$(nuevodirectorio$, poscar%, 3) = tecla$ + ":\"
                    poscar% = poscar% + 3
               ELSE
                    MID$(nuevodirectorio$, poscar%, 1) = tecla$
                    poscar% = poscar% + 1
               END IF

          CASE "I" TO "Z", "0" TO "9"
               IF poscar% = 1 THEN
                    SOUND 2987, 2
               ELSE
                    MID$(nuevodirectorio$, poscar%, 1) = tecla$
                    poscar% = poscar% + 1
               END IF

          CASE "\"
               IF poscar% = 1 THEN
                    SOUND 2987, 2
               ELSE
                    IF (MID$(nuevodirectorio$, poscar% - 1, 1) = "\") THEN
                         SOUND 2987, 2
                    ELSE
                         MID$(nuevodirectorio$, poscar%, 1) = tecla$
                         poscar% = poscar% + 1
                    END IF
               END IF

          CASE "!", "#", "$", "%", "&", "'", "(", ")", "-", "@", "^", "_", "`", "{", "}", "~"
               IF poscar% = 1 THEN
                    SOUND 2987, 2
               ELSE
                    MID$(nuevodirectorio$, poscar%, 1) = tecla$
                    poscar% = poscar% + 1
               END IF

          CASE IS = CHR$(8)                                   ' backspace
               IF poscar% = 1 THEN
                    SOUND 2987, 2                             ' cadena vacia
               ELSEIF poscar% = 4 THEN
                    poscar% = 1
                    MID$(nuevodirectorio$, poscar%, 3) = STRING$(3, 250)
               ELSE
                    poscar% = poscar% - 1
                    MID$(nuevodirectorio$, poscar%, 1) = CHR$(250)
               END IF

          CASE IS = CHR$(13)                                  ' enter
               IF poscar% = 1 THEN
                    SOUND 2987, 2                             ' cadena vacia
               ELSE
                    finuevodir% = si
               END IF
```

```
          CASE IS = CHR$(27)                                        ' escape
               escape% = si
               finuevodir% = si

          CASE ELSE
               SOUND 2987, 2

     END SELECT

     LOCATE filinicialar - 4, 30: PRINT nuevodirectorio$
     IF poscar% = fincadena% THEN
          finuevodir% = si
          SOUND 2987, 2
     ELSE
          COLOR 31, 1: LOCATE filinicialar - 4, 30 + poscar% - 1: PRINT "_";
          COLOR 15, 1
     END IF
LOOP

FOR I = 1 TO fincadena%                                             ' quitar los CHR$(250) del
     IF MID$(nuevodirectorio$, I, 1) = CHR$(250) THEN               ' final de la cadena
          nuevodirectorio$ = LEFT$(nuevodirectorio$, I - 1)
          EXIT FOR
     END IF
NEXT I
COLOR 7, 1

END SUB
```

```
'    ========================================
'    leerteclado
'    devuelve la UCASE$ de la tecla presionada
'    ========================================
SUB leerteclado (tecla$)

tecla$ = ""
DO WHILE tecla$ = ""
       tecla$ = UCASE$(INKEY$)
LOOP

END SUB



'    ================
'    nuevoabonado
'    agregar abonados
'    ================
SUB nuevoabonado (nomarchivo$)

CALL escrirot(titu, nomarchivo$, opcion%)
CALL escrirot(subt, "AGREGAR", opcion%)

numabonados% = LOF(numarchivo) / longarchivo
IF numabonados% = maxabonados THEN
     CALL direstado
     EXIT SUB
END IF

DIM temporal AS registro
DIM agregado AS registro
agregado.tipo = 10

finagr% = no
DO WHILE finagr% = no

'    *****     ingreso de nombre y numero
     numhay$ = LTRIM$(STR$(numabonados%))
     numfal$ = LTRIM$(STR$(maxabonados - numabonados%))
     numact$ = LTRIM$(STR$(numabonados% + 1))

     CALL escrimen(col1, 6, "numero de abonados en el directorio : " + STRING$((3 -
LEN(numhay$)), 32) + numhay$)
     CALL escrimen(col1, 7, "                 espacio disponible : " + STRING$((3 -
LEN(numfal$)), 32) + numfal$)
     CALL escrimen(col1, 9, "                                abonado # " + STRING$((3 -
LEN(numact$)), 32) + numact$)

     CALL ingcadena(agregado.NOMBRE, tipnoma, escape%)
     IF escape% = si THEN
          EXIT DO
     END IF

     CALL ingcadena(agregado.telefono, tiptela, escape%)
     IF escape% = si THEN
          EXIT DO
     END IF

     CALL escrimen(cen1, 13, "¨ingreso correcto? ")
     CALL escrimen(cen1, 14, "  < S > : si       ")
     CALL escrimen(cen1, 15, "  < ▒ > : modificar")
     CALL escrimen(cen1, 16, " < ESC >: salir    ")

     CALL leerteclado(opcion$)
     CALL cuadro(nulo, 13, 10, 16, 70)

     IF opcion$ = CHR$(27) THEN
          finagr% = si
     ELSEIF opcion$ = "S" THEN
```

```
'    *****     localizar posicion para nuevo abonado
          recno% = 1
          finagr2% = no
          DO WHILE (recno% <= numabonados%) AND (finagr2% = no)
               GET #numarchivo, recno%, temporal
               IF temporal.NOMBRE < agregado.NOMBRE THEN
                    recno% = recno% + 1
               ELSE
                    finagr2% = si
               END IF
          LOOP

          FOR reg% = numabonados% TO recno% STEP -1
               GET #numarchivo, reg%, temporal
               PUT #numarchivo, reg% + 1, temporal
          NEXT reg%

          PUT #numarchivo, recno%, agregado

          numabonados% = LOF(numarchivo) / longarchivo
          IF numabonados% = maxabonados THEN
               CALL direstado                                    ' si directorio lleno, no agregar
               finagr% = si
          ELSE
               CALL escrimen(cen1, 13, "  < ▓ > : agregar otro")
               CALL escrimen(cen1, 14, " < ESC >: salir       ")

               CALL leerteclado(tecla$)
               IF tecla$ = CHR$(27) THEN
                    finagr% = si
               END IF
          END IF
     END IF
LOOP

END SUB
```

```
'    ==================
'    recibir
'    recibir directorio
'    ==================
SUB recibir (nomarchivo$, cadenarx$)

DIM temporal AS registro

'    *****     obtener cadena de datos
datos$ = RTRIM$(MID$(cadenarx$, 9))
IF LEN(datos$) = 0 THEN
     SOUND 2987, 2
     CALL escrimen(cen1, 18, "LISTADO RECIBIDO VACIO")
     CALL escrimen(cen1, 19, "< ▒ >: continuar")
     nomarchivo$ = "*"
     CALL leerteclado(tecla$)
     EXIT SUB
END IF

nomarchivorx$ = LTRIM$(LEFT$(cadenarx$, 8))

'    ------------------------------------------------
'    directorio actual y listado de *.rnd que contiene
'    ------------------------------------------------
DIM listarchivos$(1 TO 100)
CLS
CALL direcyarchi(nomdirectorio$, totarchivos%, listarchivos$())
totarchivos$ = LTRIM$(STR$(totarchivos%))

CALL escrirot(titu, "RECIBIR LISTADO", opcion%)
COLOR 11, 1
CALL escrimen(cen1, filinicialar - 5, nomdirectorio$)
COLOR 7, 1

archinum% = 1
filactiva% = 1
CALL browsearchi((archinum%), filactiva%, totarchivos%, listarchivos$())
CALL cuadro(simple, filinicialar - 1, colarchivo - 4, filinicialar + totalfilasar, colarchivo +
11)

CALL escrirot(rotnomarchirx, nomarchivorx$, opcion%)
CALL escrirot(submenurx, mensaje$, opcion%)
IF totarchivos% > 1 THEN
     CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
END IF

'    --------
'    pantalla
'    --------
finrx% = no
DO WHILE finrx% = no
     IF totarchivos% > 0 THEN
          CALL escrimen(colarchivo, filinicialar + 14, LTRIM$(STR$(archinum%)) + " / " +
totarchivos$ + "   ")
     END IF

     CALL leerteclado(opcion$)

     SELECT CASE opcion$

'    *****     consultar listardo de archivos
          CASE CHR$(0) + CHR$(71), CHR$(0) + CHR$(72), CHR$(0) + CHR$(73), CHR$(0) + CHR$(79),
CHR$(0) + CHR$(80), CHR$(0) + CHR$(81)
               CALL consultarchivo(opcion$, archinum%, filactiva%, totarchivos%, listarchivos$())

'    *****     cambiar de directorio
          CASE IS = "C"
               CALL cuadro(nulo, filinicialar, col3, 20, 79)
               CALL escrimen(col3, filinicialar, "     < ESC > : salir")

               CALL ingnuevodir(nuevodirectorio$, escape%)
               IF escape% = si THEN
                    CALL escrimen(col1, filinicialar - 4, STRING$(68, 32))
                    CALL escrirot(submenurx, mensaje$, opcion%)
```

```
                    IF totarchivos% > 1 THEN
                         CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
                    END IF
               ELSE
                    ordendrive$ = LEFT$(nuevodirectorio$, 2)
                    nompath$ = MID$(nuevodirectorio$, 3)
                    ordenpath$ = "cd " + nompath$

                    CLS
                    SHELL "del ___(adt).tmp"
                    SHELL ordendrive$
                    SHELL ordenpath$
                    CLS

                    CALL direcyarchi(nomdirectorio$, totarchivos%, listarchivos$())
                    IF (MID$(nuevodirectorio$, 4) <> MID$(nomdirectorio$, 4)) THEN
                         SOUND 2987, 2
                         COLOR 12, 7
                         CALL cuadro(nulo, 3, 1, 5, 80)
                         CALL escrimen(cen1, 3, CHR$(34) + nuevodirectorio$ + CHR$(34))
                         CALL escrimen(cen1, 4, "DIRECTORIO INDICADO INEXISTENTE")
                         CALL escrimen(cen1, 5, "CAMBIO NO EFECTUADO")
                         COLOR 7, 1
                         CALL escrimen(cen1, 7, "< ▒ >: continuar")
                         CALL leerteclado(tecla$)
                    END IF

                    totarchivos$ = LTRIM$(STR$(totarchivos%))
                    CALL escrirot(titu, "RECIBIR LISTADO", opcion%)
                    COLOR 11, 1
                    CALL escrimen(cen1, filinicialar - 5, nomdirectorio$)
                    COLOR 7, 1

                    archinum% = 1
                    filactiva% = 1
                    CALL browsearchi((archinum%), filactiva%, totarchivos%, listarchivos$())
                    CALL cuadro(simple, filinicialar - 1, colarchivo - 4, filinicialar +
totalfilasar, colarchivo + 11)

                    CALL escrirot(rotnomarchirx, nomarchivorx$, opcion%)
                    CALL escrirot(submenurx, mensaje$, opcion%)
                    IF totarchivos% > 1 THEN
                         CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
                    END IF
               END IF

'    *****     cambiar nombre del listado recibido
          CASE IS = "N"
               CALL cuadro(nulo, filinicialar, col3, 20, 79)
               CALL escrimen(col3, filinicialar + 4, "     < ESC > : salir")

               CALL escrimen(col3, filinicialar, "ingresar nuevo nombre para el listado")
               CALL ingcadena(nomarchivo$, tiparch, escape%)

               IF escape% = no THEN
                    CALL escrimen(col3, filinicialar + 3, "¨ingreso correcto?")
                    CALL escrimen(col3, filinicialar + 4, " < S > : si           ")
                    CALL escrimen(col3, filinicialar + 5, " < ▒ > : no")

                    CALL leerteclado(tecla$)
                    IF tecla$ = "S" THEN
                         nomarchivorx$ = nomarchivo$
                         CALL escrirot(rotnomarchirx, nomarchivorx$ + STRING$(7, 32), opcion%)
                    END IF
               END IF

               CALL cuadro(nulo, filinicialar + 3, col3, filinicialar + 6, 79)
               CALL escrirot(submenurx, mensaje$, opcion%)
               IF totarchivos% > 1 THEN
                    CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
               END IF
```

```
'    *****     grabar listado recibido en disco
          CASE IS = "G"
               nomarchivo$ = nomarchivorx$ + ".rnd"

               IF LEN(nomarchivorx$) = 0 THEN
                    CALL escrirot(alarnopcion, "INGRESAR NOMBRE PARA EL LISTADO RECIBIDO",
opcion%)
                    CALL cuadro(simple, filinicialar - 1, colarchivo - 4, filinicialar +
totalfilasar, colarchivo + 11)
               ELSE

               yaexiste% = no
               FOR contador% = 1 TO totarchivos%
                    IF nomarchivorx$ = RTRIM$(listarchivos$(contador%)) THEN
                         yaexiste% = si
                         EXIT FOR
                    END IF
               NEXT contador%

               IF yaexiste% = no THEN
                    sigrabar% = si
               ELSE
                    SOUND 2987, 2                                        ' existe archivo
                    CALL cuadro(nulo, filinicialar, col3, 20, 79)        ' con ese nombre
                    IF totarchivos% <= totalfilasar THEN
                         archinum% = 1
                         filactiva% = contador%
                         CALL browsearchi(archinum%, filactiva%, totarchivos%, listarchivos$())
                         archinum% = contador%
                    ELSE
                         IF contador% <= totalfilasar THEN
                              archinum% = 1
                              filactiva% = contador%
                              CALL browsearchi(archinum%, filactiva%, totarchivos%,
listarchivos$())
                              archinum% = contador%
                         ELSE
                              archinum% = contador% - totalfilasar + 1
                              filactiva% = totalfilasar
                              CALL browsearchi(archinum%, filactiva%, totarchivos%,
listarchivos$())
                              archinum% = archinum% - 1
                         END IF
                    END IF
                    CALL escrimen(colarchivo, filinicialar + 14, LTRIM$(STR$(archinum%)) + " / "
+ totarchivos$ + "   ")

                    CALL escrimen(col3, filinicialar + 0, "el archivo ya existe")
                    CALL escrimen(col3, filinicialar + 1, " < R > : reemplazar")
                    CALL escrimen(col3, filinicialar + 2, " < ▒ > : invalidar ingreso")

                    CALL leerteclado(tecla$)
                    IF tecla$ = "R" THEN
                         KILL nomarchivo$
                         sigrabar% = si
                    ELSE
                         sigrabar% = no
                         CALL cuadro(nulo, filinicialar + 3, col3, filinicialar + 6, 79)
                         CALL escrirot(submenurx, mensaje$, opcion%)
                         IF totarchivos% > 1 THEN
                              CALL escrirot(rotbrowsearchi, mensaje$, opcion%)
                         END IF
                    END IF
               END IF

               IF sigrabar% = si THEN
                    OPEN nomarchivo$ FOR RANDOM AS #numarchivo LEN = longarchivo

                    CALL escrirot(titu, nomarchivo$, opcion%)
                    CALL escrirot(subt, "CARGAR DIRECTORIO DE TRABAJO", opcion%)
                    CALL escrimen(cen1, 7, "conversion de formato")

                    numabonados% = LEN(datos$) / byblo
                    numabonados$ = LTRIM$(STR$(numabonados%))
```

```
                    FOR recno% = 1 TO numabonados%
                         CALL escrimen(ceni, 8, "   [ " + LTRIM$(STR$(recno%)) + " / " +
numabonados$ + " ]   ")

                         temporal.NOMBRE = MID$(datos$, (recno% - 1) * byblo + 1, 16)

                         puntero% = 1
                         FOR num% = 1 TO 7
                              numbyte% = (recno% - 1) * byblo + 16 + num%

                              msn% = ASC(MID$(datos$, numbyte%, 1)) \ 16
                              IF msn% = 10 THEN
                                   IF puntero% = 1 THEN
                                        msn$ = CHR$(48)
                                        MID$(temporal.telefono, puntero%, 1) = msn$
                                        puntero% = puntero% + 1
                                   ELSE
                                        msn$ = CHR$(32)
                                   END IF
                              ELSE
                                   msn$ = CHR$(48 + msn%)
                              END IF
                              MID$(temporal.telefono, puntero%, 1) = msn$
                              puntero% = puntero% + 1

                              lsn% = ASC(MID$(datos$, numbyte%, 1)) MOD 16
                              IF num% = 7 THEN
                                   temporal.tipo = lsn%
                              ELSE
                                   IF lsn% = 10 THEN
                                        lsn$ = CHR$(32)
                                   ELSE
                                        lsn$ = CHR$(48 + lsn%)
                                   END IF
                                   MID$(temporal.telefono, puntero%, 1) = lsn$
                                   puntero% = puntero% + 1
                              END IF
                         NEXT num%

                         IF puntero% = 14 THEN
                              MID$(temporal.telefono, puntero%, 1) = CHR$(32)
                         END IF

                         PUT #numarchivo, recno%, temporal
                    NEXT recno%

                    CALL escrirot(alarnopcion, "conversion OK", 9)
                    finrx% = si
               END IF

               END IF

'    *****     escape
          CASE IS = CHR$(27)
               CALL escrimen(colarchivo, filinicialar + 14, STRING$(8, 32))
               CALL escrirot(alar, "SALIR DE RECEPCION", finrx%)
               IF finrx% = si THEN
                    nomarchivo$ = "*"
               ELSE
                 .  CALL cuadro(simple, filinicialar - 1, colarchivo - 4, filinicialar +
totalfilasar, colarchivo + 11)
               END IF


          CASE ELSE
               SOUND 2987, 2

     END SELECT
LOOP

END SUB
```

```
'    ============================
'    transmitir
'    transmitir listado telefonico
'    ============================
SUB transmitir (nomarchivo$)

CALL escrirot(titu, nomarchivo$, opcion%)
CALL escrirot(subt, "TRANSMITIR LISTADO TELEFONICO", opcion%)
CALL escrirot(mencomunica, mensaje$, opcion%)
CALL escrirot(alarnopcion, "listo para transmitir", teclanovae%)
CALL cuadro(nulo, 6, 2, 13, 79)

IF LOF(numarchivo) = 0 THEN
     CALL direstado
     EXIT SUB
END IF

DIM temporal AS registro

numabonados% = LOF(numarchivo) / longarchivo
numabonados$ = LTRIM$(STR$(numabonados%))

CALL escrimen(cen1, 7, "generando informacion a transmitir")
OPEN "com1:9600,n,8,1,bin,cd0,cs0,ds0,op0,rs" FOR RANDOM AS #numpuerto

'    *****     establecer comunicacion con el equipo
PRINT #numpuerto, careqrx;

retardoi& = TIMER                                   ' retardo de 1 a 2 seg
finretardo% = no
DO WHILE finretardo% = no
     retardof& = TIMER
     IF retardof& = retardoi& + 2 THEN
          finretardo% = si
     END IF
LOOP
CALL escrimen(cen1, 7, "          TRANSMITIENDO           ")

'    *****     transmitir nombre del achivo
nomarchise$ = LEFT$(nomarchivo$, (LEN(nomarchivo$) - 4))
nomarchise$ = STRING$(8 - LEN(nomarchise$), 32) + nomarchise$
PRINT #numpuerto, nomarchise$;
CALL escrimen(cen1, 10, "nombre del archivo: OK")

'    *****     transmision del directorio, conversion a formato del equipo
DIM nibbletelef%(15)

FOR recno% = 1 TO numabonados%
     CALL escrimen(cen1, 12, "   [ " + LTRIM$(STR$(recno%)) + " / " + numabonados$ + " ]    ")
     GET #numarchivo, recno%, temporal

     telefndc$ = STRING$(7, 170)
     FOR I = 1 TO 14
          IF MID$(temporal.telefono, I, 1) = CHR$(32) THEN
               nibbletelef%(I) = 10
          ELSE
               nibbletelef%(I) = VAL(MID$(temporal.telefono, I, 1))
          END IF
     NEXT I

     IF (nibbletelef%(1) = 0 AND nibbletelef%(2) = 0) THEN
          nibbletelef%(2) = 10
          puntero% = 2
          nibbletelef%(15) = temporal.tipo
     ELSE
          puntero% = 1
          nibbletelef%(14) = temporal.tipo
     END IF

     FOR I = 1 TO 7
          MID$(telefndc$, I, 1) = STRING$(1, (16 * nibbletelef%(puntero%) + nibbletelef%(puntero%
+ 1)))
          puntero% = puntero% + 2
```

```
    NEXT I

    PRINT #numpuerto, temporal.NOMBRE; telefndc$;

NEXT recno%

PRINT #numpuerto, carfindir;
CALL escrimen(cen1, 13, "directorio : OK")
CLOSE #numpuerto
CALL escrirot(alarnopcion, "TRANSMISION TERMINADA", opcion%)

END SUB
```

```
'         ================================================
'         PROGRAMA PARA ADMINISTRAR DIRECTORIOS TELEFONICOS
'                        modulo MENU&HLP.BAS
'                            96/05/06
'
'              - menues
'              - ayudas
'         ================================================


DECLARE SUB cuadro (tipcuad%, fili%, coli%, filf%, colf%)
DECLARE SUB escrimen (tipmen%, fila%, mensaje$)
DECLARE SUB escrirot (tiprot%, mensaje$, opcion%)

DECLARE SUB menupri (numenu%, habilmp%, menuinici%)
DECLARE SUB menupop (numenu%, sileerteclado%, opcion$, finmen%, menuinici%)
DECLARE SUB flechamp (numenu%, nupop%)
DECLARE SUB leerteclado (tecla$)

CONST colmp1 = 2                    ' columna para popmenu 1: Archivo
CONST colmp2 = 14                   ' columna para popmenu 2: Comunicacion
CONST colmp3 = 31                   ' columna para popmenu 3: Edicion
CONST colmp4 = 65                   ' columna para popmenu 4: Ayuda

CONST linmenayu = 38                ' longitud de una linea del mensaje de ayuda
CONST filayu = 16                   ' fila para el recuadro de ayuda
CONST filmp = 3                     ' fila para inicio de popmenu

CONST nulo = 0                      ' limpia el interior del cuadro
CONST simple = 1                    ' cuadro simple
CONST doble = 2                     ' cuadro doble

CONST cen1 = 101                    ' mensaje centrado en pantalla
CONST colabout = 8                  ' columna para Help/General

CONST alar = 1                      ' rotulo es mensaje de alarma
CONST titu = 3                      ' rotulo es titulo

CONST si = -1                       ' .T.
CONST no = 0                        ' .F.
```

```
'    ================================
'    about
'    informacion general del programa
'    ================================
SUB about

CLS
COLOR 14, 1
CALL cuadro(doble, 1, 1, 25, 80)

COLOR 11, 1
CALL escrimen(cen1, 3, "ADITEL-pc96")
CALL escrimen(cen1, 4, "ADMINISTRADOR DE DIRECTORIOS TELEFONICOS")

COLOR 7, 1
LOCATE 7, colabout: PRINT "Programa para generar y editar listados telefonicos, desarrollado"
LOCATE 8, colabout: PRINT "como elemento complementario al equipo administrador de un"
LOCATE 9, colabout: PRINT "directorio electronico para dos lineas telefonicas, ";
COLOR 15, 1: PRINT "ADITEL-eq96";
COLOR 7, 1: PRINT "."

LOCATE 11, colabout + 3: PRINT "Autor:       Henderson Iturralde."
LOCATE 12, colabout + 3: PRINT "Proyecto:    ";
COLOR 15, 1: PRINT "ADITEL-pc96";
COLOR 7, 1: PRINT " forma parte de la tesis de grado"
LOCATE 13, colabout + 3: PRINT "             previa a la obtencion del titulo de Ingeniero"
LOCATE 14, colabout + 3: PRINT "             en Electronica y Telecomunicaciones del autor."
LOCATE 15, colabout + 3: PRINT "Director:    Ing. Fabio Gonz lez."
LOCATE 16, colabout + 3: PRINT "Institucion: Escuela Politecnica Nacional."
LOCATE 17, colabout + 3: PRINT "             Facultad de Ingenieria Electrica."
LOCATE 18, colabout + 3: PRINT "Fecha:       Quito, 9 de abril de 1996."

CALL escrimen(cen1, 20, "team Ronda")
CALL escrimen(cen1, 21, "(c) 1993 - 1996")
CALL escrimen(cen1, 23, "< ▓ >: presione cualquier tecla para continuar")
CALL leerteclado(manualgen$)

END SUB
```

```
'    ==================
'    manual
'    manual del usuario
'    ==================
SUB manual

CLS
CALL escrirot(titu, "manual del usuario", opcion%)
COLOR 14, 1
CALL escrimen(1, 20, CHR$(199) + STRING$(78, 196) + CHR$(182))
COLOR 7, 1

CALL escrimen(cen1, 21, " < I > : indice                < 1 >...< 9 >: ir a la p gina  ")
CALL escrimen(cen1, 22, " < F7 >: imprimir manual        < Page Up > : p gina anterior ")
CALL escrimen(cen1, 23, "< ESC >: salir                 < Page Down >: p gina siguiente")

pagina% = 0
sileerteclado% = no
opcion$ = "I"
finman% = no
DO WHILE finman% = no
     IF sileerteclado% = si THEN
          CALL leerteclado(opcion$)
     END IF
     sileerteclado% = si

     SELECT CASE opcion$
          CASE IS = "1"
               pagina% = 1
               CALL cuadro(nulo, 3, 5, 18, 79)
               LOCATE 3, 8:  PRINT "p gina 1"
               LOCATE 5, 8:  COLOR 15, 1: PRINT "Mensajes en pantalla.": COLOR 7, 1
               LOCATE 6, 5:  PRINT "- En la linea superior de la pantalla aparecer  el nombre del
programa"
               LOCATE 7, 5:  PRINT "  junto con el nombre del archivo de trabajo e inmediatamente
abajo, la"
               LOCATE 8, 5:  PRINT "  funcion que se est  ejecutando en ese momento"
               LOCATE 9, 5:  PRINT "- Las teclas que el usuario puede presionar para ejecutar una
funcion"
               LOCATE 10, 5: PRINT "  aparecer n resaltadas y entre los '< >', seguidos de una
breve"
               LOCATE 11, 5: PRINT "  descripcion de dicha funcion."
               LOCATE 12, 5: PRINT "- Se utiliza el signo de trebol: < ▒ >, para indicar que
presionando"
               LOCATE 13, 5: PRINT "  cualquier tecla (o cualquier otra tecla) se ejecuta la
accion"
               LOCATE 14, 5: PRINT "  indicada."

          CASE IS = "2"
               pagina% = 2
               CALL cuadro(nulo, 3, 5, 18, 79)
               LOCATE 3, 8:  PRINT "p gina 2"
               LOCATE 5, 8:  COLOR 15, 1: PRINT "Barras de menus.": COLOR 7, 1
               LOCATE 6, 5:  PRINT "Las funciones que el programa puede realizar han sido
agrupadas, segun"
               LOCATE 7, 5:  PRINT "su car cter, en menus, y estos, a su vez, en barras de
menus."
               LOCATE 8, 5:  PRINT "Hay dos barras de menus:"
               LOCATE 9, 5:  PRINT "- Barra de menus de inicializacion:"
               LOCATE 10, 5: PRINT "  Aparece al iniciar el programa o al cambiar o recibir un
archivo de"
               LOCATE 11, 5: PRINT "  trabajo. Permite abrir un archivo de trabajo o recibir un
listado"
               LOCATE 12, 5: PRINT "  telefonico del equipo."
               LOCATE 13, 5: PRINT "- Barra de menus principal:"
               LOCATE 14, 5: PRINT "  Aparece cuando ya se cuenta con un archivo de trabajo.
Permite"
               LOCATE 15, 5: PRINT "  modificar, consultar o imprimir los datos del archivo,
comunicarse"
               LOCATE 16, 5: PRINT "  con el equipo."

          CASE IS = "3"
               pagina% = 3
               CALL cuadro(nulo, 3, 5, 18, 79)
```

```
                LOCATE 3, 8:  PRINT "p gina 3"
                LOCATE 5, 8:  COLOR 15, 1: PRINT "Seleccion de un menu o funcion.": COLOR 7, 1
                LOCATE 6, 5:  PRINT "La seleccion de cualquier funcion dentro de un menu, o de un
menu"
                LOCATE 7, 5:  PRINT "dentro de una barra, se puede realizar presionando la tecla"
                LOCATE 8, 5:  PRINT "correspondiente a la letra resaltada de la palabra o,
posicion ndose"
                LOCATE 9, 5:  PRINT "con las teclas de navegacion (flechas) sobre la opcion y
luego"
                LOCATE 10, 5: PRINT "presionando < ENTER >."


          CASE IS = "4"
                pagina% = 4
                CALL cuadro(nulo, 3, 5, 18, 79)
                LOCATE 3, 8:  PRINT "p gina 4"
                LOCATE 5, 8:  COLOR 15, 1: PRINT "Menu ARCHIVO (barra de inicializacion).": COLOR
7, 1
                LOCATE 6, 5:  PRINT "Presenta las opeciones:"
                LOCATE 7, 5:  PRINT "- Abrir / crear."
                LOCATE 8, 5:  PRINT "  Cambia de drive o subdirectorio de trabajo, abre un
archivo"
                LOCATE 9, 5:  PRINT "  existente o crea uno nuevo."
                LOCATE 10, 5: PRINT "- Recibir."
                LOCATE 11, 5: PRINT "  Recibe el listado telefonico almacenado en la memoria del
equipo."
                LOCATE 12, 5: PRINT "- Salir."
                LOCATE 13, 5: PRINT "  Sale del programa."
                LOCATE 16, 5: PRINT "Importante: Antes de seleccionar la opcion 'Recibir',
serciorese de que"
                LOCATE 17, 5: PRINT "            el cable de transmision/recepcion est  conectado
como se"
                LOCATE 18, 5: PRINT "            indica en el manual del equipo."

          CASE IS = "5"
                pagina% = 5
                CALL cuadro(nulo, 3, 5, 18, 79)
                LOCATE 3, 8:  PRINT "p gina 5"
                LOCATE 5, 8:  COLOR 15, 1: PRINT "Menu ARCHIVO (barra principal).": COLOR 7, 1
                LOCATE 6, 5:  PRINT "Presenta las opciones:"
                LOCATE 7, 5:  PRINT "- Abrir / crear."
                LOCATE 8, 5:  PRINT "  Cambia de drive o subdirectorio de trabajo, abre un
archivo"
                LOCATE 9, 5:  PRINT "  existente o crea uno nuevo."
                LOCATE 10, 5: PRINT "- Consultar."
                LOCATE 11, 5: PRINT "  Muestra en una ventana el contenido del archivo de trabajo
(solo si"
                LOCATE 12, 5: PRINT "  si este contiene datos)."
                LOCATE 13, 5: PRINT "- Imprimir."
                LOCATE 14, 5: PRINT "  Imprime el contenido del archivo de trabajo (solo si este
contiene"
                LOCATE 15, 5: PRINT "  datos."
                LOCATE 16, 5: PRINT "- Salir."
                LOCATE 17, 5: PRINT "  Sale del programa."

          CASE IS = "6"
                pagina% = 6
                CALL cuadro(nulo, 3, 5, 18, 79)
                LOCATE 3, 8:  PRINT "p gina 6"
                LOCATE 5, 8:  COLOR 15, 1: PRINT "Menu COMUNICACION.": COLOR 7, 1
                LOCATE 6, 5:  PRINT "Presenta las opciones:"
                LOCATE 7, 5:  PRINT "- Recibir."
                LOCATE 8, 5:  PRINT "  Recibe el listado telefonico almacenado en la memoria del
equipo."
                LOCATE 9, 5:  PRINT "- Transmitir."
                LOCATE 10, 5: PRINT "  Envia al equipo el listado telefonico contenido en el
archivo de"
                LOCATE 11, 5: PRINT "  trabajo."
                LOCATE 16, 5: PRINT "Importante: Antes de seleccionar cualquiera de estas
opciones,"
                LOCATE 17, 5: PRINT "            serciorese de que el cable de
transmision/recepcion est "
                LOCATE 18, 5: PRINT "            conectado como se indica en el manual del
equipo."
```

```
          CASE IS = "7"
               pagina% = 7
               CALL cuadro(nulo, 3, 5, 18, 79)
               LOCATE 3, 8:  PRINT "p gina 7"
               LOCATE 5, 8:  COLOR 15, 1: PRINT "Menu EDITAR.": COLOR 7, 1
               LOCATE 6, 5:  PRINT "Presenta las opciones:"
               LOCATE 7, 5:  PRINT "- Borrar."
               LOCATE 8, 5:  PRINT "  Borra un abonado del listado telefonico."
               LOCATE 9, 5:  PRINT "- Definir."
               LOCATE 10, 5: PRINT "  Define a los abonados como normales, abreviados o
comnutados"
               LOCATE 11, 5: PRINT "  directos."
               LOCATE 12, 5: PRINT "- Nuevo."
               LOCATE 13, 5: PRINT "  Agrega un nuevo abonado al listado telefonico. Los datos
pedidos"
               LOCATE 14, 5: PRINT "  ser n el nombre del abonado y su numero telefonico."
               LOCATE 15, 5: PRINT "  * El nombre tiene una longitud de 16 caracteres y solo
puede constar"
               LOCATE 16, 5: PRINT "    de letras, A a Z, o espacios en blanco (excepto al
inicio)."
               LOCATE 17, 5: PRINT "  * El numero tiene una extension de 14 digitos, 0 a 9."

          CASE IS = "8"
               pagina% = 8
               CALL cuadro(nulo, 3, 5, 18, 79)
               LOCATE 3, 8:  PRINT "p gina 8"
               LOCATE 5, 8:  COLOR 15, 1: PRINT "Menu HELP.": COLOR 7, 1
               LOCATE 6, 5:  PRINT "Presenta las opciones:"
               LOCATE 7, 5:  PRINT "- General."
               LOCATE 8, 5:  PRINT "  Muestra en pantalla una informacion general del programa."
               LOCATE 9, 5:  PRINT "- Manual."
               LOCATE 10, 5: PRINT "  Muestra en pantalla un resumen del manual del usuario y
permite su"
               LOCATE 11, 5: PRINT "  impresion."

          CASE IS = "9"
               pagina% = 9
               CALL cuadro(nulo, 3, 5, 18, 79)
               LOCATE 3, 8:  PRINT "p gina 9"
               LOCATE 5, 8:  COLOR 15, 1: PRINT "Errores.": COLOR 7, 1
               LOCATE 6, 5:  PRINT "El programa es capaz de administrar cualquier error que se
presente"
               LOCATE 7, 5:  PRINT "durante su ejecucion, dando un tratamiento especial a los
errores de"
               LOCATE 8, 5:  PRINT "impresion, cambio de drive y de disco protegido contra
escritura; y"
               LOCATE 9, 5:  PRINT "permitiendo al usuario escoger entre varias opciones. En todo
caso,"
               LOCATE 10, 5: PRINT "se pueden evitar los errores m s frecuentes considerando las"
               LOCATE 11, 5: PRINT "siguientes instrucciones:"
               LOCATE 12, 5: PRINT "- Errores de impresion."
               LOCATE 13, 5: PRINT "  Asegurese de que la impresora est  bien conectada al
computador,"
               LOCATE 14, 5: PRINT "  que est  encendida y que se halla cargado el papel
necesario."
               LOCATE 15, 5: PRINT "- Errores en cambio de drive y disco."
               LOCATE 16, 5: PRINT "  Antes de realizar un cambio de drive, asegurese de que el
disco est "
               LOCATE 17, 5: PRINT "  insertado en el drive y que no se halla protegido contra
escritura."

          CASE IS = CHR$(0) + CHR$(73)                          ' Page Up
               sileerteclado% = no
               IF pagina% = 0 THEN
                    opcion$ = "9"
               ELSE
                    pagina% = pagina% - 1
                    IF pagina% < 1 THEN
                         opcion$ = "9"
                    ELSE
                         opcion$ = CHR$(48 + pagina%)
                    END IF
               END IF
```

```
CASE IS = CHR$(0) + CHR$(81)                              ' Page Down
     sileerteclado% = no
     IF pagina% = 0 THEN
          opcion$ = "1"
     ELSE
          pagina% = pagina% + 1
          IF pagina% > 9 THEN
               opcion$ = "1"
          ELSE
               opcion$ = CHR$(48 + pagina%)
          END IF
     END IF

CASE IS = "I"
     pagina% = 0
     CALL cuadro(nulo, 3, 5, 18, 79)
     COLOR 15, 1
     CALL escrimen(cen1, 4, "INDICE")
     COLOR 7, 1
     CALL escrimen(cen1, 6, "< 1 >:   Mensajes en pantalla.                  ")
     CALL escrimen(cen1, 7, "< 2 >:   Barras de menus.                       ")
     CALL escrimen(cen1, 8, "< 3 >:   Seleccion de un menu o funcion.        ")
     CALL escrimen(cen1, 9, "< 4 >:   Menu ARCHIVO (barra de inicializacion).")
     CALL escrimen(cen1, 10, "< 5 >:   Menu ARCHIVO (barra principal).        ")
     CALL escrimen(cen1, 11, "< 6 >:   Menu COMUNICACION.                     ")
     CALL escrimen(cen1, 12, "< 7 >:   Menu EDITAR.                           ")
     CALL escrimen(cen1, 13, "< 8 >:   Menu HELP.                             ")
     CALL escrimen(cen1, 14, "< 9 >:   Errores.                               ")

CASE IS = CHR$(27)
     finman% = si

CASE IS = CHR$(0) + CHR$(65)                              ' F7: imprimir
     CALL cuadro(nulo, 3, 5, 18, 79)
     sileerteclado% = no
     opcion$ = "I"

     SOUND 2987, 2
     CALL escrimen(cen1, 10, "< F7 >: imprimir manual del usuario")
     CALL escrimen(cen1, 11, " < ▒ >: continual                   ")
     CALL leerteclado(simprime$)
     IF simprime$ = CHR$(0) + CHR$(65) THEN
          CALL cuadro(nulo, 10, 2, 11, 79)
          CALL escrimen(cen1, 5, "imprimiendo")

          LPRINT "ADITEL-pc96"
          LPRINT "ADMINISTRADOR DE DIRECTORIOS TELEFONICOS"
          LPRINT
          LPRINT "Programa para generar y editar listados telefonicos, desarrollado"
          LPRINT "como elemento complementario al equipo administrador de un"
          LPRINT "directorio electronico para dos lineas telefonicas, ADITEL - eq96."
          LPRINT
          LPRINT "Autor:       Henderson Iturralde."
          LPRINT "Proyecto:    ADITEL - pc96 forma parte de la tesis de grado"
          LPRINT "             previa a la obtencion del titulo de Ingeniero"
          LPRINT "             en Electronica y Telecomunicaciones del autor."
          LPRINT "Director:    Ing. Fabio Gonz lez."
          LPRINT "Institucion: Escuela Politecnica Nacional."
          LPRINT "             Facultad de Ingenieria Electrica."
          LPRINT "Fecha:       Quito, 9 de abril de 1996."
          LPRINT
          LPRINT "team Ronda"
          LPRINT "(c) 1993 - 1996"
          LPRINT
          LPRINT
          LPRINT
          LPRINT "     INDICE"
          LPRINT "1:   Mensajes en pantalla."
          LPRINT "2:   Barras de menus."
          LPRINT "3:   Seleccion de un menu o funcion."
          LPRINT "4:   Menu ARCHIVO (barra de inicializacion)."
          LPRINT "5:   Menu ARCHIVO (barra principal)."
          LPRINT "6:   Menu COMUNICACION."
```

```
                LPRINT "7:   Menu EDITAR."
                LPRINT "8:   Menu HELP."
                LPRINT "9:   Errores."
                LPRINT CHR$(12)

                LPRINT " 1.- Mensajes en pantalla."
                LPRINT
                LPRINT "- En la linea superior de la pantalla aparecer  el nombre del
programa"
                LPRINT "  junto con el nombre del archivo de trabajo e inmediatamente abajo,
la"
                LPRINT "  funcion que se est  ejecutando en ese momento"
                LPRINT "- Las teclas que el usuario puede presionar para ejecutar una
funcion"
                LPRINT "  aparecer n resaltadas y entre los '< >', seguidos de una breve"
                LPRINT "  descripcion de dicha funcion."
                LPRINT "- Se utiliza el signo de trebol: < ▒ >, para indicar que presionando"
                LPRINT "  cualquier tecla (o cualquier otra tecla) se ejecuta la accion"
                LPRINT "  indicada."
                LPRINT
                LPRINT
                LPRINT " 2.- Barras de menus."
                LPRINT "Las funciones que el programa puede realizar han sido agrupadas,
segun"
                LPRINT "su car cter, en menus, y estos, a su vez, en barras de menus."
                LPRINT "Hay dos barras de menus:"
                LPRINT "- Barra de menus de inicializacion:"
                LPRINT "  Aparece al iniciar el programa o al cambiar o recibir un archivo
de"
                LPRINT "  trabajo. Permite abrir un archivo de trabajo o recibir un listado"
                LPRINT "  telefonico del equipo."
                LPRINT "- Barra de menus principal:"
                LPRINT "  Aparece cuando ya se cuenta con un archivo de trabajo. Permite"
                LPRINT "  modificar, consultar o imprimir los datos del archivo, comunicarse"
                LPRINT "  con el equipo."
                LPRINT
                LPRINT
                LPRINT " 3.- Seleccion de un menu o funcion."
                LPRINT "La seleccion de cualquier funcion dentro de un menu, o de un menu"
                LPRINT "dentro de una barra, se puede realizar presionando la tecla"
                LPRINT "correspondiente a la letra resaltada de la palabra o, posicion ndose"
                LPRINT "con las teclas de navegacion (flechas) sobre la opcion y luego"
                LPRINT "presionando < ENTER >."
                LPRINT
                LPRINT
                LPRINT " 4.- Menu ARCHIVO (barra de inicializacion)."
                LPRINT "Presenta las opeciones:"
                LPRINT "- Abrir / crear."
                LPRINT "  Cambia de drive o subdirectorio de trabajo, abre un archivo"
                LPRINT "  existente o crea uno nuevo."
                LPRINT "- Recibir."
                LPRINT "  Recibe el listado telefonico almacenado en la memoria del equipo."
                LPRINT "- Salir."
                LPRINT "  Sale del programa."
                LPRINT "Importante: Antes de seleccionar la opcion 'Recibir', serciorese de
que"
                LPRINT "            el cable de transmision/recepcion est  conectado como se"
                LPRINT "            indica en el manual del equipo."
                LPRINT
                LPRINT
                LPRINT " 5.- Menu ARCHIVO (barra principal)."
                LPRINT "Presenta las opciones:"
                LPRINT "- Abrir / crear."
                LPRINT "  Cambia de drive o subdirectorio de trabajo, abre un archivo"
                LPRINT "  existente o crea uno nuevo."
                LPRINT "- Consultar."
                LPRINT "  Muestra en una ventana el contenido del archivo de trabajo (solo
si"
                LPRINT "  si este contiene datos)."
                LPRINT "- Imprimir."
                LPRINT "  Imprime el contenido del archivo de trabajo (solo si este contiene"
                LPRINT "  datos."
                LPRINT "- Salir."
                LPRINT "  Sale del programa."
```

```
                    LPRINT
                    LPRINT
                    LPRINT " 6.- Menu COMUNICACION."
                    LPRINT "Presenta las opciones:"
                    LPRINT "- Recibir."
                    LPRINT "  Recibe el listado telefonico almacenado en la memoria del equipo."
                    LPRINT "- Transmitir."
                    LPRINT "  Envia al equipo el listado telefonico contenido en el archivo de"
                    LPRINT "  trabajo."
                    LPRINT "Importante: Antes de seleccionar cualquiera de estas opciones,"
                    LPRINT "            serciorese de que el cable de transmision/recepcion est "
                    LPRINT "            conectado como se indica en el manual del equipo."
                    LPRINT
                    LPRINT
                    LPRINT " 7.- Menu EDITAR."
                    LPRINT "Presenta las opciones:"
                    LPRINT "- Borrar."
                    LPRINT "  Borra un abonado del listado telefonico."
                    LPRINT "- Definir."
                    LPRINT "  Define a los abonados como normales, abreviados o comnutados"
                    LPRINT "  directos."
                    LPRINT "- Nuevo."
                    LPRINT "  Agrega un nuevo abonado al listado telefonico. Los datos pedidos"
                    LPRINT "  ser n el nombre del abonado y su numero telefonico."
                    LPRINT "  * El nombre tiene una longitud de 16 caracteres y solo puede
constar"
                    LPRINT "    de letras, A a Z, o espacios en blanco (excepto al inicio)."
                    LPRINT "  * El numero tiene una extension de 14 digitos, 0 a 9."
                    LPRINT
                    LPRINT
                    LPRINT " 8.- Menu HELP."
                    LPRINT "Presenta las opciones:"
                    LPRINT "- General."
                    LPRINT "  Muestra en pantalla una informacion general del programa."
                    LPRINT "- Manual."
                    LPRINT "  Muestra en pantalla un resumen del manual del usuario y permite su"
                    LPRINT "  impresion."
                    LPRINT
                    LPRINT
                    LPRINT " 9.- Errores."
                    LPRINT "El programa es capaz de administrar cualquier error que se presente"
                    LPRINT "durante su ejecucion, dando un tratamiento especial a los errores de"
                    LPRINT "impresion, cambio de drive y de disco protegido contra escritura; y"
                    LPRINT "permitiendo al usuario escoger entre varias opciones. En todo caso,"
                    LPRINT "se pueden evitar los errores m s frecuentes considerando las"
                    LPRINT "siguientes instrucciones:"
                    LPRINT "- Errores de impresion."
                    LPRINT "  Asegurese de que la impresora est  bien conectada al computador,"
                    LPRINT "  que est  encendida y que se halla cargado el papel necesario."
                    LPRINT "- Errores en cambio de drive y disco."
                    LPRINT "  Antes de realizar un cambio de drive, asegurese de que el disco
est "
                    LPRINT "  insertado en el drive y que no se halla protegido contra
escritura."
                    LPRINT CHR$(12)

                    SOUND 2987, 2
               END IF

          CASE ELSE
               SOUND 2987, 2

     END SELECT
LOOP

END SUB
```

```
'    ==========================
'    menayuda
'    mensajes de ayuda en linea
'    ==========================
SUB menayuda (numayuda%)

SELECT CASE numayuda%
     CASE IS = 1, 11
          mensaje1$ = "Abrir / nuevo"
          mensaje2$ = "Especificar el drive y directorio de"
          mensaje3$ = "trabajo y abrir un archivo existente"
          mensaje4$ = "o crear uno nuevo."

     CASE IS = 2, 21
          mensaje1$ = "Recibir      "
          mensaje2$ = "Recibir el listado telefonico"
          mensaje3$ = "almacenado en la memoria del equipo y"
          mensaje4$ = "guardarlo en un archivo."

     CASE IS = 3, 14
          mensaje1$ = "Salir        "
          mensaje2$ = "Salir del programa."
          mensaje3$ = ""
          mensaje4$ = ""

     CASE IS = 12
          mensaje1$ = "Consultar    "
          mensaje2$ = "Consultar el listado telefonico."
          mensaje3$ = ""
          mensaje4$ = ""

     CASE IS = 13
          mensaje1$ = "Imprimir     "
          mensaje2$ = "Imprimir el listado telefonico."
          mensaje3$ = ""
          mensaje4$ = ""

     CASE IS = 22
          mensaje1$ = "Transmitir   "
          mensaje2$ = "Transmitir el listado telefonico"
          mensaje3$ = "contenido en el archivo de trabajo a"
          mensaje4$ = "la memoria del equipo."

     CASE IS = 31
          mensaje1$ = "Borrar       "
          mensaje2$ = "Borrar un abonado del listado"
          mensaje3$ = "telefonico."
          mensaje4$ = ""

     CASE IS = 32
          mensaje1$ = "Definir      "
          mensaje2$ = "Definir abonados normales,"
          mensaje3$ = "abreviados y conmutados directos."
          mensaje4$ = ""

     CASE IS = 33
          mensaje1$ = "Nuevo        "
          mensaje2$ = "Agregar un abonado nuevo al listado"
          mensaje3$ = "telefonico."
          mensaje4$ = ""

     CASE IS = 41
          mensaje1$ = "General      "
          mensaje2$ = "Informacion general del programa."
          mensaje3$ = ""
          mensaje4$ = ""

     CASE IS = 42
          mensaje1$ = "Manual       "
          mensaje2$ = "Indicar en pantalla el manual del"
          mensaje3$ = "usuario y permitir su impresion."
          mensaje4$ = ""

END SELECT
```

```
COLOR 12, 7
mensaje1$ = " " + mensaje1$ + " "
CALL escrimen(40 - (linmenayu / 2 - 2), filayu + 0, mensaje1$)
COLOR 0, 7

mensaje2$ = mensaje2$ + STRING$(linmenayu - LEN(mensaje2$), 32)
CALL escrimen(cen1, filayu + 2, mensaje2$)

mensaje3$ = mensaje3$ + STRING$(linmenayu - LEN(mensaje3$), 32)
CALL escrimen(cen1, filayu + 3, mensaje3$)

mensaje4$ = mensaje4$ + STRING$(linmenayu - LEN(mensaje4$), 32)
CALL escrimen(cen1, filayu + 4, mensaje4$)

COLOR 7, 1

END SUB
```

```
'    ==================================
'    menu
'    menu de inicializacion y principal
'    ==================================
SUB menu (nomarchivo$, menuinici%, opcion$)

DIM mpri$(0 TO 4)
mpri$(0) = "A"
mpri$(1) = "A"
mpri$(2) = "C"
mpri$(3) = "E"
mpri$(4) = "H"

IF menuinici% = si THEN
     CALL escrirot(titu, "INICIALIZACION", opcion%)
     numenu% = 0
ELSE
     CALL escrirot(titu, nomarchivo$, opcion%)
     numenu% = 1
END IF
LOCATE 2, 2: COLOR 0, 7: PRINT STRING$(78, 32);

habilmp% = no
CALL menupri(numenu%, habilmp%, menuinici%)

finmen% = no
sileerteclado% = si
DO WHILE finmen% = no
     IF sileerteclado% = si THEN
          CALL leerteclado(opcion$)
     END IF
     sileerteclado% = si

     SELECT CASE opcion$
          CASE IS = CHR$(0) + CHR$(75)                                    ' LEFT
               numenu% = numenu% - 1
               CALL menupri(numenu%, habilmp%, menuinici%)

          CASE IS = CHR$(0) + CHR$(77)                                    ' RIGHT
               numenu% = numenu% + 1
               CALL menupri(numenu%, habilmp%, menuinici%)

          CASE IS = CHR$(0) + CHR$(72), CHR$(0) + CHR$(80), CHR$(13)      ' UP/DOWN/ENTER
               opcion$ = mpri$(numenu%)
               sileerteclado% = no

          CASE IS = "A"                                                   ' Archivo
               IF menuinici% = si THEN
                    numenu% = 0
               ELSE
                    numenu% = 1
               END IF
               CALL menupop(numenu%, sileerteclado%, opcion$, finmen%, menuinici%)


          CASE IS = "C"                                                   ' Comunicacion
               numenu% = 2
               CALL menupop(numenu%, sileerteclado%, opcion$, finmen%, menuinici%)

          CASE IS = "E"                                                   ' Editar
               numenu% = 3
               CALL menupop(numenu%, sileerteclado%, opcion$, finmen%, menuinici%)

          CASE IS = "H"                                                   ' Help
               numenu% = 4
               CALL menupop(numenu%, sileerteclado%, opcion$, finmen%, menuinici%)

          CASE ELSE
               SOUND 2987, 2

     END SELECT
```

```
     IF ((finmen% = si) AND (opcion$ = "A" OR opcion$ = "R" OR opcion$ = "T" OR opcion$ = "S"))
THEN
          SELECT CASE opcion$
               CASE IS = "A"
                    CALL escrirot(alar, "ABRIR / CREAR ARCHIVO DE TRABAJO", finmen%)

               CASE IS = "R"
                    CALL escrirot(alar, "RECIBIR LISTADO TELEFONICO DEL EQUIPO", finmen%)

               CASE IS = "T"
                    CALL escrirot(alar, "TRANSMITIR LISTADO TELEFONICO AL EQUIPO", finmen%)

               CASE IS = "S"
                    CALL escrirot(alar, "SALIR DEL PROGRAMA", finmen%)

          END SELECT

          IF finmen% = no THEN
               CALL cuadro(nulo, filmp, 2, filmp + 8, 79)
               CALL menupri(numenu%, habilmp%, menuinici%)
               sileerteclado% = si
          END IF
     END IF
LOOP

END SUB
```

```
'    *****     popmenu ARCHIVO (de menu principal)
     CASE IS = 1
          DIM mpop$(4)
          mpop$(1) = "A"
          mpop$(2) = "C"
          mpop$(3) = "I"
          mpop$(4) = "S"

          LOCATE filmp + 0, colmp1: COLOR 7, 1: PRINT "³   "; : COLOR 15, 1: PRINT "A"; : COLOR
7, 1: PRINT "brir / nuevo   ³";
          LOCATE filmp + 1, colmp1: COLOR 7, 1: PRINT "³  ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ  ³";
          LOCATE filmp + 2, colmp1: COLOR 7, 1: PRINT "³   "; : COLOR 15, 1: PRINT "C"; : COLOR
7, 1: PRINT "onsultar       ³";
          LOCATE filmp + 3, colmp1: COLOR 7, 1: PRINT "³   "; : COLOR 15, 1: PRINT "I"; : COLOR
7, 1: PRINT "mprimir        ³";
          LOCATE filmp + 4, colmp1: COLOR 7, 1: PRINT "³  ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ  ³";
          LOCATE filmp + 5, colmp1: COLOR 7, 1: PRINT "³   "; : COLOR 15, 1: PRINT "S"; : COLOR
7, 1: PRINT "alir           ³";
          LOCATE filmp + 6, colmp1: COLOR 7, 1: PRINT "ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ";

          DO WHILE finpop% = no
               CALL flechamp(numenu%, nupop%)
               CALL leerteclado(opcion$)

               SELECT CASE opcion$
                    CASE mpop$(1), mpop$(2), mpop$(3), mpop$(4)
                         finmen% = si
                         finpop% = si

                    CASE IS = CHR$(0) + CHR$(72)                    ' UP
                         nupop% = nupop% - 1

                    CASE IS = CHR$(0) + CHR$(80)                    ' DOWN
                         nupop% = nupop% + 1

                    CASE IS = CHR$(0) + CHR$(75)                    ' LEFT
                         CALL cuadro(nulo, filmp, colmp1, filmp + 8, colmp1 + 20)
                         opcion$ = "H"
                         finpop% = si

                    CASE IS = CHR$(0) + CHR$(77)                    ' RIGHT
                         CALL cuadro(nulo, filmp, colmp1, filmp + 8, colmp1 + 20)
                         opcion$ = "C"
                         finpop% = si

                    CASE IS = CHR$(13)                              ' ENTER
                         opcion$ = mpop$(nupop%)
                         finmen% = si
                         finpop% = si

                    CASE IS = CHR$(27)                              ' ESCAPE
                         CALL cuadro(nulo, filmp, colmp1, filmp + 8, colmp1 + 20)
                         CALL menupri(numenu%, habilmp%, menuinici%)
                         finpop% = si
                         sileerteclado% = si

                    CASE ELSE
                         SOUND 2987, 2

               END SELECT
          LOOP

'    *****     popmenu COMUNICACION
     CASE IS = 2
          DIM mpop$(2)
          mpop$(1) = "R"
          mpop$(2) = "T"

          LOCATE filmp + 0, colmp2: COLOR 7, 1: PRINT "³   "; : COLOR 15, 1: PRINT "R"; : COLOR
7, 1: PRINT "ecepcion     ³";
          LOCATE filmp + 1, colmp2: COLOR 7, 1: PRINT "³   "; : COLOR 15, 1: PRINT "T"; : COLOR
7, 1: PRINT "ransmision   ³";
          LOCATE filmp + 2, colmp2: COLOR 7, 1: PRINT "ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ";
```



```
                    nupop% = nupop% + 1
```

```
                    CASE IS = CHR$(0) + CHR$(75)                        ' LEFT
                         CALL cuadro(nulo, filmp, colmp3, filmp + 3, colmp3 + 14)
                         opcion$ = "C"
                         finpop% = si

                    CASE IS = CHR$(0) + CHR$(77)                        ' RIGHT
                         CALL cuadro(nulo, filmp, colmp3, filmp + 3, colmp3 + 14)
                         opcion$ = "H"
                         finpop% = si

                    CASE IS = CHR$(13)                                  ' ENTER
                         opcion$ = mpop$(nupop%)
                         finmen% = si
                         finpop% = si

                    CASE IS = CHR$(27)                                  ' ESCAPE
                         CALL cuadro(nulo, filmp, colmp3, filmp + 3, colmp3 + 14)
                         CALL menupri(numenu%, habilmp%, menuinici%)
                         finpop% = si
                         sileerteclado% = si

                    CASE ELSE
                         SOUND 2987, 2

               END SELECT
          LOOP

'    *****     popmenu HELP
     CASE IS = 4
          DIM mpop$(2)
          mpop$(1) = "G"
          mpop$(2) = "M"

          LOCATE filmp + 0, colmp4: COLOR 7, 1: PRINT "³   "; : COLOR 15, 1: PRINT "G"; : COLOR
7, 1: PRINT "eneral   ³";
          LOCATE filmp + 1, colmp4: COLOR 7, 1: PRINT "³   "; : COLOR 15, 1: PRINT "M"; : COLOR
7, 1: PRINT "anual    ³";
          LOCATE filmp + 2, colmp4: COLOR 7, 1: PRINT "ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ";

          DO WHILE finpop% = no
               CALL flechamp(numenu%, nupop%)
               CALL leerteclado(opcion$)

               SELECT CASE opcion$
                    CASE mpop$(1), mpop$(2)
                         finmen% = si
                         finpop% = si

                    CASE IS = CHR$(0) + CHR$(72)                        ' UP
                         nupop% = nupop% - 1

                    CASE IS = CHR$(0) + CHR$(80)                        ' DOWN
                         nupop% = nupop% + 1

                    CASE IS = CHR$(0) + CHR$(75)                        ' LEFT
                         CALL cuadro(nulo, filmp, colmp4, filmp + 2, colmp4 + 14)
                         IF menuinici% = si THEN
                              opcion$ = "A"
                         ELSE
                              opcion$ = "E"
                         END IF
                         finpop% = si

                    CASE IS = CHR$(0) + CHR$(77)                        ' RIGHT
                         CALL cuadro(nulo, filmp, colmp4, filmp + 2, colmp4 + 14)
                         opcion$ = "A"
                         finpop% = si

                    CASE IS = CHR$(13)                                  ' ENTER
                         opcion$ = mpop$(nupop%)
                         finmen% = si
                         finpop% = si
```

```
                    CASE IS = CHR$(27)                                  ' ESCAPE
                         CALL cuadro(nulo, filmp, colmp4, filmp + 2, colmp4 + 14)
                         CALL menupri(numenu%, habilmp%, menuinici%)
                         finpop% = si
                         sileerteclado% = si

                    CASE ELSE
                         SOUND 2987, 2

               END SELECT
          LOOP

END SELECT

IF NOT (finmen% = no AND (opcion$ = "A" OR opcion$ = "C" OR opcion$ = "E" OR opcion$ = "H")) THEN
     COLOR 7, 1
     CALL cuadro(nulo, filayu, 40 - (linmenayu / 2 + 3), filayu + 6, 40 + (linmenayu / 2 + 3))
END IF

END SUB
```

```
'     ========================================================
'     menupri
'     presentar barra del menu de inicializacion o principal
'     ========================================================
SUB menupri (numenu%, habilmp%, menuinici%)

IF menuinici% = si THEN
     IF numenu% = -1 THEN
          numenu% = 4
     ELSEIF numenu% = 1 THEN
          numenu% = 4
     ELSEIF numenu% = 3 THEN
          numenu% = 0
     ELSEIF numenu% = 5 THEN
          numenu% = 0
     END IF
ELSE
     IF numenu% = 0 THEN
          numenu% = 4
     ELSEIF numenu% = 5 THEN
          numenu% = 1
     END IF
END IF

LOCATE 2, colmp1 + 3
IF (numenu% = 0 OR numenu% = 1) THEN
     COLOR 14, 1: PRINT " Archivo ";
ELSE
     IF habilmp% = si THEN
          COLOR 0, 7: PRINT " Archivo ";
     ELSE
          COLOR 15, 7: PRINT " A"; : COLOR 0, 7: PRINT "rchivo ";
     END IF
END IF

IF menuinici% = no THEN
     LOCATE 2, colmp2 + 3
     IF numenu% = 2 THEN
          COLOR 14, 1: PRINT " Comunicacion ";
     ELSE
          IF habilmp% = si THEN
               COLOR 0, 7: PRINT " Comunicacion ";
          ELSE
               COLOR 15, 7: PRINT " C"; : COLOR 0, 7: PRINT "omunicacion ";
          END IF
     END IF

     LOCATE 2, colmp3 + 3
     IF numenu% = 3 THEN
          COLOR 14, 1: PRINT " Editar ";
          COLOR 0, 7
     ELSE
          IF habilmp% = si THEN
               COLOR 0, 7: PRINT " Editar ";
          ELSE
               COLOR 15, 7: PRINT " E"; : COLOR 0, 7: PRINT "ditar ";
          END IF
     END IF
END IF

LOCATE 2, colmp4'+ 3
IF numenu% = 4 THEN
     COLOR 14, 1: PRINT " Help ";
     COLOR 0, 7
ELSE
     IF habilmp% = si THEN
          COLOR 0, 7: PRINT " Help ";
     ELSE
          COLOR 15, 7: PRINT " H"; : COLOR 0, 7: PRINT "elp ";
     END IF
END IF

END SUB
```