```c
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

#include <windows.h>

/*---------------------------------------------------------------------------*
 * Codigos de retorno:
 *---------------------------------------------------------------------------*/
#define  EXIT_SUCCESS        0
#define  EXIT_FAILURE        1

/*---------------------------------------------------------------------------*
 * Maximos:
 *---------------------------------------------------------------------------*/
#define  NUM_COLORES        16
#define  MAX_BUFFER       2048

/*---------------------------------------------------------------------------*
 * Codigos de error:
 *---------------------------------------------------------------------------*/
#define  PIX_E1              1
#define  PIX_E2              2
#define  PIX_E3              3
#define  PIX_E4              4
#define  PIX_E5              5

/*---------------------------------------------------------------------------*
 * Variables globales:
 *---------------------------------------------------------------------------*/
char *NomProg;     /* Nombre del programa en minusculas.                     */
char *DirPrograma; /* Directorio del programa y los ficheros auxiliares.     */
char  Color[NUM_COLORES] =
     {'#',    // VGA_NEGRO         0  negro
      'r',    // VGA_ROJO          4  rojo
      'V',    // VGA_VERDE         2  verde,verdeoscuro
      's',    // VGA_MARRON        6  marron,sepia,oroviejo
      'A',    // VGA_AZULOSCURO    1  azuloscuro
      'M',    // VGA_MAGENTA       5  violeta,morado
      'c',    // VGA_CYAN          3  cyan,a¤il
      '-',    // VGA_GRISOSCURO    8  gris
      '.',    // VGA_GRIS          7  grisclaro
      'n',    // VGA_NARANJA      12  naranja
      'v',    // VGA_VERDECLARO   10  verdeclaro
      'o',    // VGA_AMARILLO     14  amarillo
      'a',    // VGA_AZUL          9  azul
      'm',    // VGA_VIOLETACLARO 13  violetaclaro
      'C',    // VGA_AZULCLARO    11  azulclaro
      'ú' }; // VGA_BLANCO       15  blanco

/*---------------------------------------------------------------------------*/
   int main(int argc, char **argv)
/* LIBRERIA: Pix
 * EDICION: 10 de Agosto de 1994
 * SINOPSIS: Convierte un fichero BMP (Windows) a formato ICO.
 * DESCRIPCION: Convierte un fichero Bitmap Windows (*.BMP) al formato
 *   propio ICO (*.ICO), que luego puede compilarse mediante SAM y
 *   convertirse a formato PixMap (*.ICO).
 *   Este formato ICO es el siguiente:
```

```
 *   !---------------------------------------------------------------
 *   ! Definicion del icono: icoEqLogo.
 *   !---------------------------------------------------------------
 *   Tabla de colores:
 *     Azul oscuro= 'a'
 *     Rojo       = 'r'
 *     Gris claro = 'g'
 *   Fin de la tabla de colores.
 *   Definici¢n del pixmap:
 *   aaaaaaa . . . rrrrrrrrrrrrrrr . . .
 *   aaaaaaa . . . rrrrgggrrrrrrrr . . .
 *   aaaaaaa . . . rrrrrrrrrrrggrr . . .
 *   Fin del pixmap.
 *
 *   - Cabecera: Tabla de colores.
 *   - Cuerpo:   Matriz de caracteres.
 *   - Restricciones:
 *     - El BMP debe ser de 16 colores.
 *     - ((Ancho * Alto) / 2) < (64*1024) - 16 // El tama¤o de la imagen,
 *       medio byte por pixel mas lo que ocupa la cabecera (se reservan 16
 *       bytes) debe ser menor de 64 Kbytes, para que quepa en un segmento
 *       del anticuado DOS.
 *     - Antiguamente, el Ancho del bitmap debia ser multiplo de 32 pixels,
 *       para facilitar el tratamiento. Actualmente esta restriccion no
 *       existe.
 *
 * EJEMPLO: Bmp2Ico Origen Destino
 *
 * CONSULTAR: Pix_Muestra(), Bmp2Pix (ejecutable).
 * NOTAS: Para generar ficheros BMP a partir de cualquier otro formato
 *   puede utilizarse el programa Alchemy, indicandoles que el formato
 *   destino es BMP (opcion -w) a 16 colores (opcion -c16).
 *     alchemy -w -c16 origen destino
 *   Se recomienda utilizar el editor de bitmaps de Windows para comprobar
 *   el resultado y salir salvando, de esta forma se normaliza la paleta
 *   de colores.
 *
 * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
 *-------------------------------------------------------------------------*/
{
int               Arg;                      // Para procesar argumentos.
int               X, Y, XMax, YMax, I, H, L;
long              RowBytes, RowStart;
char             *Pos;                      // Para procesar strings.
char              NomBmp[MAX_BUFFER],
                  NomPix[MAX_BUFFER];       // Nombre de los ficheros.
FILE             *Bmp, *Pix;                // Controladores de ficheros.
BITMAPFILEHEADER  Bmfh;
BITMAPINFOHEADER  Bmih;
RGBQUAD           RGBColores[NUM_COLORES];
BYTE              BitmapBits;
static char       Buffer[MAX_BUFFER];
int               NErr=EXIT_SUCCESS;        // Para retornar a DOS.
int               Pix_Help();               // Se declaran despues.
int               Pix_Copyright();
int               Pix_Error(char *, int);
int               Pix_Verbose(char*, char*, int, int);

if ((Pos = strrchr((NomProg=strlwr (*argv)),'\\')) != NULL)
  {
```

```
121     DirPrograma = *argv;
122     *Pos++ = '\0';
123     NomProg = Pos;
124     }
125
126   if ((Pos = strrchr(NomProg,'.')) != NULL) { *Pos = '\0'; }
127
128   Pix_Copyright();
129
130   if (argc != 3) { NErr=Pix_Help(); }
131   else
132     {
133     strncpy(NomBmp, argv[1], sizeof NomBmp);
134     if ((Pos=strchr(NomBmp,'.'))==NULL) { strcat(NomBmp,".BMP"); }
135     if ((Bmp=fopen(NomBmp,"rb"))==NULL) { NErr=Pix_Error(NomBmp,PIX_E1); }
136     else
137       {
138       strncpy(NomPix, argv[2], sizeof NomPix);
139       if ((Pos=strchr(NomPix,'.'))==NULL) { strcat(NomPix,".ICO"); }
140       if ((Pix=fopen(NomPix,"w"))==NULL) { NErr=Pix_Error(NomPix,PIX_E2); }
141       else
142         {
143         fread (&Bmfh, sizeof(BITMAPFILEHEADER), 1, Bmp);
144         if (*(int *)"BM" != Bmfh.bfType) { NErr=Pix_Error(NomBmp,PIX_E3); }
145         else
146           {
147           fread (&Bmih, sizeof(BITMAPINFOHEADER), 1, Bmp);
148           if (Bmih.biBitCount != 4) { NErr=Pix_Error(NomBmp,PIX_E4); }
149           else
150             {
151             fread (&RGBColores, sizeof RGBColores, 1, Bmp);
152             I = 0;
153             RowBytes = ((((Bmih.biWidth + 1) >> 1) + 3) >> 2) << 2;
154             RowStart = Bmfh.bfOffBits + (Bmih.biHeight - 1) * RowBytes;
155
156             Pix_Verbose(NomBmp, NomPix, Bmih.biWidth, Bmih.biHeight);
157             fprintf(Pix,"\n");
158             fprintf(Pix,"\n!------------------------------------------------");
159             fprintf(Pix,"\n! Definicion del icono: %s,",NomBmp);
160             fprintf(Pix,"\n!------------------------------------------------");
161             fprintf(Pix,"\nTabla de colores:");
162             fprintf(Pix,"\n  Negro        = '%c'",Color[ 0]);
163             fprintf(Pix,"\n  Rojo         = '%c'",Color[ 1]);
164             fprintf(Pix,"\n  VerdeOscuro  = '%c'",Color[ 2]);
165             fprintf(Pix,"\n  Marron       = '%c'",Color[ 3]);
166             fprintf(Pix,"\n  AzulOscuro   = '%c'",Color[ 4]);
167             fprintf(Pix,"\n  Violeta      = '%c'",Color[ 5]);
168             fprintf(Pix,"\n  Cyan         = '%c'",Color[ 6]);
169             fprintf(Pix,"\n  Gris         = '%c'",Color[ 7]);
170             fprintf(Pix,"\n  GrisClaro    = '%c'",Color[ 8]);
171             fprintf(Pix,"\n  Naranja      = '%c'",Color[ 9]);
172             fprintf(Pix,"\n  VerdeClaro   = '%c'",Color[10]);
173             fprintf(Pix,"\n  Amarillo     = '%c'",Color[11]);
174             fprintf(Pix,"\n  Azul         = '%c'",Color[12]);
175             fprintf(Pix,"\n  VioletaClaro = '%c'",Color[13]);
176             fprintf(Pix,"\n  AzulClaro    = '%c'",Color[14]);
177             fprintf(Pix,"\n  Blanco       = '%c'",Color[15]);
178             fprintf(Pix,"\nFin de la tabla de colores.");
179             fprintf(Pix,"\nDefinici¢n del pixmap:");
180             fprintf(Pix,"\n");
```

```
181
182             for (Y=0, YMax=Bmih.biHeight; Y < YMax; Y++)
183               {
184               fseek(Bmp, RowStart, SEEK_SET);
185               for (X=0, XMax=Bmih.biWidth; X < XMax; X++)
186                 {
187                 Arg = fgetc(Bmp);
188                 H = (Arg >> 4) & 0x0F;  H = Color[H];
189                 L = Arg & 0x0F;         L = Color[L];
190                 X++;
191                 fprintf(Pix,"%c%c",(char)H,(char)L);
192                 }
193               RowStart -= RowBytes;
194               fprintf(Pix,"\n");
195               fputc('.',stdout); fflush(stdout);
196               }
197             }
198           }
199         fprintf(Pix,"Fin del pixmap.\n\n");
200         fclose(Pix);
201         }
202       fclose(Bmp);
203       }
204     }
205   printf("\n");
206
207   return(NErr);
208   }
209
210   /*--------------------------------------------------------------------------*/
211      int Pix_Copyright()
212   /* LIBRERIA: Pix
213    * EDICION: 10 de Agosto de 1994
214    * SINOPSIS: Visualiza la autoria del conversor.
215    *
216    * EJEMPLO: Pix_Copyright()
217    *
218    * CONSULTAR: Pix_Error(), Pix_Verbose().
219    *
220    * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
221    *--------------------------------------------------------------------------*/
222   {
223   printf("\nENTORNO McEQ");
224   printf("\n%s: Conversor de ficheros BMP (Windows) a ICO (Pixmaps)",NomProg);
225   printf("\n(c) 1994 QRC Informatica, EQ Sistemas Inteligentes, V.1.0");
226
227   return (EXIT_SUCCESS);
228   }
229
230   /*--------------------------------------------------------------------------*/
231      int Pix_Verbose(char *Origen, char *Destino, int Ancho, int Alto)
232   /* LIBRERIA: Pix
233    * EDICION: 10 de Agosto de 1994
234    * SINOPSIS: Visualiza el trabajo de conversion que se va a realizar.
235    *
236    * EJEMPLO: Pix_Copyright()
237    *
238    * CONSULTAR: Pix_Error(), Pix_Help().
239    *
240    * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
```

```
241   *-------------------------------------------------------------------------*/
242   {
243   printf("\nConversion: de %s[X=%d,Y=%d] a %s[X=%d,Y=%d]\n",
244             Origen, Ancho, Alto, Destino, Ancho, Alto);
245
246   return (EXIT_SUCCESS);
247   }
248
249
250   /*-------------------------------------------------------------------------*/
251      int Pix_Help()
252   /* LIBRERIA: Pix
253    * EDICION: 10 de Agosto de 1994
254    * SINOPSIS: Visualiza la ayuda del conversor de formatos.
255    *
256    * EJEMPLO: Pix_Help()
257    *
258    * CONSULTAR: Pix_Error(), Pix_Verbose().
259    *
260    * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
261    *-------------------------------------------------------------------------*/
262   {
263   fprintf(stderr,"\nUso: %s Origen[.BMP] Destino[.ICO]",NomProg);
264
265   return (EXIT_FAILURE);
266   }
267
268   /*-------------------------------------------------------------------------*/
269      int Pix_Error(char *Nombre, int Codigo)
270   /* LIBRERIA: Pix
271    * EDICION: 10 de Agosto de 1994
272    * SINOPSIS: Visualiza los erres del conversor de formatos.
273    *
274    * EJEMPLO: Pix_Error(NomBmp,PIX_E1)
275    *
276    * CONSULTAR: Pix_Help().
277    *
278    * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
279    *-------------------------------------------------------------------------*/
280   {
281   fprintf(stderr,"\nERROR: %d, ",Codigo);
282   if       (Codigo == PIX_E1)
283     { fprintf(stderr,"%s: No puede abrir su entrada [%s].",NomProg,Nombre); }
284   else if (Codigo == PIX_E2)
285     { fprintf(stderr,"%s: No puede crear su salida [%s].",NomProg,Nombre); }
286   else if (Codigo == PIX_E3)
287     { fprintf(stderr,"%s: El archivo [%s] no es tipo BMP.",NomProg,Nombre); }
288   else if (Codigo == PIX_E4)
289     { fprintf(stderr,"%s: [%s] no es un BMP de %2d colores.",
290                      NomProg,Nombre,NUM_COLORES); }
291   else
292     { fprintf(stderr,"%s: Error interno desconocido.",NomProg); }
293
294   fprintf(stderr,"\n");
295
296   return (EXIT_FAILURE);
297   }
298
299
```

```
1
2
3   !-------------------------------------------
4   ! Definicion del icono: asolver.bmp,
5   !-------------------------------------------
6   Tabla de colores:
7     Negro        = '#'
8     Rojo         = 'r'
9     VerdeOscuro  = 'V'
10    Marron       = 's'
11    AzulOscuro   = 'A'
12    Violeta      = 'M'
13    Cyan         = 'c'
14    Gris         = '°'
15    GrisClaro    = '.'
16    Naranja      = 'n'
17    VerdeClaro   = 'v'
18    Amarillo     = 'o'
19    Azul         = 'a'
20    VioletaClaro = 'm'
21    AzulClaro    = 'C'
22    Blanco       = '-'
23  Fin de la tabla de colores.
24  Definici¢n del pixmap:
25  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn.----------.°nnnnnnnnnnnnnnnnnnnnnnnn
26  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn°---------.nnnnnnnnnnnnnnnnnnnnnnnnn
27  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn---------nnnnnnnnnnnnnnnnnnnnnnnnnn
28  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn--------.nnnnnnnnnnnnnnnnnnnnnnnnnn
29  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnn°--------.nnnnnnnnnnnnnnnnnnnnnnnnnn
30  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn--------.nnnnnnnnnnnnnnnnnnnnnnnnnn
31  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn.--------.nnnnnnnnnnnnnnnnnnnnnnnnnn
32  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn----------°nnnnnnnnnnnnnnnnnnnnnnnnn
33  nnnnnnnnnnnnnnnnnnnn°°nnnnnnnnnn----------...°°nnnnnnnnnnnnnnnnnnnn
34  nnnnnnnnnnnnnn..------------..nnnn.--------------------..nnnnnnnnnnnnn
35  nnnnnnnnnnn.---------------------.nnn.----------------------.nnnnnnnnnn
36  nnnnnnnnn-------------------------.nnn.----------------------nnnnnnnnn
37  nnnnnnn.----------------------------.nnnnn...-----------------.nnnnnnn
38  nnnnnn.----------.nnnnnnnn.----------.nnnnnnnnnnnnnn.----------.nnnnnn
39  nnnnn.---------nnnnnnnnnnnnn.---------.nnnnnnnnnnnnnnn---------.nnnnn
40  nnnnn---------nnnnnnnnnnnnnnn---------nnnnnnnnnnnnnnnn---------nnnnn
41  nnnn.--------°nnnnnnnnnnnnnnn.--------°nnnnnnnnnnnnnnnn--------.nnnn
42  nnnn.--------nnnnnnnnnnnnnnnn.--------.nnnnnnnnnnnnnnnn--------.nnnn
43  nnnn°--------.nnnnnnnnnnnnnnn.--------.nnnnnnnnnnnnnnn.--------.nnnn
44  nnnnn---------.nnnnnnnnnnnnn.----------.nnnnnnnnnnnnn.---------nnnnn
45  nnnnnn----------°nnnnnnnnnn.--------------.nnnnnnnnn°----------°nnnnn
46  nnnnnnn----------.......--------------------.......----------°nnnnnn
47  nnnnnnnn.-------------------------------------------------.nnnnnnnn
48  nnnnnnnnnn.----------------------.nn.----------------------.nnnnnnnnnn
49  nnnnnnnnnnnn.-----------------..nnnnnn..-----------------.nnnnnnnnnnnn
50  nnnnnnnnnnnnnnnn....----...°nnnnnnnnnnnnnn°...-----...nnnnnnnnnnnnnnnn
51  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn
52  nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn
53  Fin del pixmap.
54
55
```

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>

#include <windows.h>

/*---------------------------------------------------------------------------*
 * Codigos de retorno:
 *---------------------------------------------------------------------------*/
#define  EXIT_SUCCESS        0
#define  EXIT_FAILURE        1

/*---------------------------------------------------------------------------*
 * Maximos:
 *---------------------------------------------------------------------------*/
#define  NUM_COLORES        16
#define  MAX_BUFFER       2048

/*---------------------------------------------------------------------------*
 * Codigos de error:
 *---------------------------------------------------------------------------*/
#define  BIF_E1              1
#define  BIF_E2              2
#define  BIF_E3              3
#define  BIF_E4              4
#define  BIF_E5              5

/*---------------------------------------------------------------------------*
 * Variables globales:
 *---------------------------------------------------------------------------*/
char *NomProg;     /* Nombre del programa en minusculas.                     */
char *DirPrograma; /* Directorio del programa y los ficheros auxiliares.     */
char  Color[NUM_COLORES] =
      { 0, 4, 2, 6, 1, 5, 3, 8, 7, 0xC, 0xA, 0xE, 9, 0xD, 0xB, 0xF };

/*---------------------------------------------------------------------------*/
   int main(int argc, char **argv)
/* LIBRERIA: Bif
 * EDICION: 10 de Agosto de 1994
 * SINOPSIS: Convierte un fichero BMP (Windows) a formato BIF.
 * DESCRIPCION: Convierte un fichero Bitmap Windows (*.BMP) al formato
 *   propio Binary Image File (*.BIF). Este formato BIF es el siguiente:
 *   - Cabecera: [BIF,XXX,YYY] // Nombre, Ancho (999), Alto (999).
 *   - Cuerpo:   BBBBBBB.....B // Cada byte B codifica 2 colores de una
 *     paleta b sica de 16 colores (parte alta y parte baja).
 *   - Restricciones:
 *     - El BMP debe ser de 16 colores.
 *     - ((Ancho * Alto) / 2) < (64*1024) - 16 // El tama¤o de la imagen,
 *       medio byte por pixel mas lo que ocupa la cabecera (se reservan 16
 *       bytes) debe ser menor de 64 Kbytes, para que quepa en un segmento
 *       del anticuado DOS.
 *     - Antiguamente, el Ancho del bitmap debia ser multiplo de 32 pixels,
 *       para facilitar el tratamiento. Actualmente esta restriccion no
 *       existe.
 *
 * EJEMPLO: Bmp2Bif Origen Destino
 *
 * CONSULTAR: Bif_Muestra(), Bmp2Pix (ejecutable).
 * NOTAS: Para generar ficheros BMP a partir de cualquier otro formato
```

```
61    *   puede utilizarse el programa Alchemy, indicandoles que el formato
62    *   destino es BMP (opcion -w) a 16 colores (opcion -c16).
63    *     alchemy -w -c16 origen destino
64    *   Se recomienda utilizar el editor de bitmaps de Windows para comprobar
65    *   el resultado y salir salvando, de esta forma se normaliza la paleta
66    *   de colores.
67    *
68    * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
69    *-------------------------------------------------------------------------*/
70   {
71   int               Arg;                        // Para procesar argumentos.
72   int               X, Y, XMax, YMax, I, H, L;
73   long              RowBytes, RowStart;
74   char             *Pos;                        // Para procesar strings.
75   char              NomBmp[MAX_BUFFER],
76                     NomBif[MAX_BUFFER];         // Nombre de los ficheros.
77   FILE             *Bmp, *Bif;                  // Controladores de ficheros.
78   BITMAPFILEHEADER  Bmfh;
79   BITMAPINFOHEADER  Bmih;
80   RGBQUAD           RGBColores[NUM_COLORES];
81   BYTE              BitmapBits;
82   static char       Buffer[MAX_BUFFER];
83   int               NErr=EXIT_SUCCESS;          // Para retornar a DOS.
84   int               Bif_Help();                 // Se declaran despues.
85   int               Bif_Copyright();
86   int               Bif_Error(char *, int);
87   int               Bif_Verbose(char*, char*, int, int);
88
89   if ((Pos = strrchr((NomProg=strlwr (*argv)),'\\')) != NULL)
90     {
91     DirPrograma = *argv;
92     *Pos++ = '\0';
93     NomProg = Pos;
94     }
95
96   if ((Pos = strrchr(NomProg,'.')) != NULL) { *Pos = '\0'; }
97
98   Bif_Copyright();
99
100  if (argc != 3) { NErr=Bif_Help(); }
101  else
102    {
103    strncpy(NomBmp, argv[1], sizeof NomBmp);
104    if ((Pos=strchr(NomBmp,'.'))==NULL) { strcat(NomBmp,".BMP"); }
105    if ((Bmp=fopen(NomBmp,"rb"))==NULL) { NErr=Bif_Error(NomBmp,BIF_E1); }
106    else
107      {
108      strncpy(NomBif, argv[2], sizeof NomBif);
109      if ((Pos=strchr(NomBif,'.'))==NULL) { strcat(NomBif,".BIF"); }
110      if ((Bif=fopen(NomBif,"wb"))==NULL) { NErr=Bif_Error(NomBif,BIF_E2); }
111      else
112        {
113        fread (&Bmfh, sizeof(BITMAPFILEHEADER), 1, Bmp);
114        if (*(int *)"BM" != Bmfh.bfType) { NErr=Bif_Error(NomBmp,BIF_E3); }
115        else
116          {
117          fread (&Bmih, sizeof(BITMAPINFOHEADER), 1, Bmp);
118          if (Bmih.biBitCount != 4) { NErr=Bif_Error(NomBmp,BIF_E4); }
119          else
120            {
```

```
121             fread (&RGBColores, sizeof RGBColores, 1, Bmp);
122             I = 0;
123             RowBytes = ((((Bmih.biWidth + 1) >> 1) + 3) >> 2) << 2;
124             RowStart = Bmfh.bfOffBits + (Bmih.biHeight - 1) * RowBytes;
125
126             Bif_Verbose(NomBmp, NomBif, Bmih.biWidth, Bmih.biHeight);
127             fprintf(Bif,"[BIF,%3d,%3d]",(int) Bmih.biWidth,
128                                         (int)Bmih.biHeight);
129
130             for (Y=0, YMax=Bmih.biHeight; Y < YMax; Y++)
131               {
132               fseek(Bmp, RowStart, SEEK_SET);
133               for (X=0, XMax=Bmih.biWidth; X < XMax; X++)
134                 {
135                 Arg = fgetc(Bmp);
136                 H = (Arg >> 4) & 0x0F;  H = Color[H];
137                 L = Arg & 0x0F;         L = Color[L];
138                 Arg = (H << 4) + L;
139                 fputc(Arg, Bif);  X++;
140                 }
141               RowStart -= RowBytes;
142               fputc('.',stdout); fflush(stdout);
143               }
144             }
145           }
146         fclose(Bif);
147         }
148       fclose(Bmp);
149       }
150     }
151   printf("\n");
152
153   return(NErr);
154   }
155
156   /*-------------------------------------------------------------------------*/
157      int Bif_Copyright()
158   /* LIBRERIA: Bif
159    * EDICION: 10 de Agosto de 1994
160    * SINOPSIS: Visualiza la autoria del conversor.
161    *
162    * EJEMPLO: Bif_Copyright()
163    *
164    * CONSULTAR: Bif_Error(), Bif_Verbose().
165    *
166    * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
167    *-------------------------------------------------------------------------*/
168   {
169   printf("\nENTORNO McEQ");
170   printf("\n%s: Conversor de ficheros BMP (Windows) a BIF",NomProg);
171   printf("\n(c) 1994 QRC Informatica, EQ Sistemas Inteligentes, V.1.0");
172
173   return (EXIT_SUCCESS);
174   }
175
176   /*-------------------------------------------------------------------------*/
177      int Bif_Verbose(char *Origen, char *Destino, int Ancho, int Alto)
178   /* LIBRERIA: Bif
179    * EDICION: 10 de Agosto de 1994
180    * SINOPSIS: Visualiza el trabajo de conversion que se va a realizar.
```

```
 *
 * EJEMPLO: Bif_Copyright()
 *
 * CONSULTAR: Bif_Error(), Bif_Help().
 *
 * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
 *-------------------------------------------------------------------------*/
{
printf("\nConversion: de %s[X=%d,Y=%d] a %s[X=%d,Y=%d]\n",
          Origen, Ancho, Alto, Destino, Ancho, Alto);

return (EXIT_SUCCESS);
}


/*-------------------------------------------------------------------------*/
   int Bif_Help()
/* LIBRERIA: Bif
 * EDICION: 10 de Agosto de 1994
 * SINOPSIS: Visualiza la ayuda del conversor de formatos.
 *
 * EJEMPLO: Bif_Help()
 *
 * CONSULTAR: Bif_Error(), Bif_Verbose().
 *
 * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
 *-------------------------------------------------------------------------*/
{
fprintf(stderr,"\nUso: %s Origen[.BMP] Destino[.BIF]",NomProg);

return (EXIT_FAILURE);
}

/*-------------------------------------------------------------------------*/
   int Bif_Error(char *Nombre, int Codigo)
/* LIBRERIA: Bif
 * EDICION: 10 de Agosto de 1994
 * SINOPSIS: Visualiza los erres del conversor de formatos.
 *
 * EJEMPLO: Bif_Error(NomBmp,BIF_E1)
 *
 * CONSULTAR: Bif_Help().
 *
 * (C) 1994 QRC Informatica, EQ Sistemas Inteligentes - V.1.0
 *-------------------------------------------------------------------------*/
{
fprintf(stderr,"\nERROR: %d, ",Codigo);
if      (Codigo == BIF_E1)
  { fprintf(stderr,"%s: No puede abrir su entrada [%s].",NomProg,Nombre); }
else if (Codigo == BIF_E2)
  { fprintf(stderr,"%s: No puede crear su salida [%s].",NomProg,Nombre); }
else if (Codigo == BIF_E3)
  { fprintf(stderr,"%s: El archivo [%s] no es tipo BMP.",NomProg,Nombre); }
else if (Codigo == BIF_E4)
  { fprintf(stderr,"%s: [%s] no es un BMP de %2d colores.",
                   NomProg,Nombre,NUM_COLORES); }
else
  { fprintf(stderr,"%s: Error interno desconocido.",NomProg); }

fprintf(stderr,"\n");
```

```
241
242   return (EXIT_FAILURE);
243   }
244
245
```