/* ———————————————————————————————— *

src2tex* version 2.12 (Sep 9, 1996)

Kazuo Amano† and Shinichi Nomoto‡

* ———————————————————————————————— */

/* ———————————— text2tex.c ———————————— */

```
#include <stdio.h>
#include "src2tex.h"

extern int TXT_flag;
extern int BAS_flag;
extern int C_flag;
extern int CBL_flag;
extern int F77_flag;
extern int HTML_flag;
extern int JAVA_flag;
extern int LISP_flag;
extern int MAKE_flag;
extern int PAS_flag;
extern int PERL_flag;
extern int SH_flag;
extern int TCL_flag;

extern int ASR_flag;
extern int MAC_flag;
extern int MAP_flag;
extern int MAT_flag;
extern int MLAB_flag;
extern int MPAD_flag;
extern int RED_flag;

extern int Page_Len_Max;
extern int Htab_Size;
extern int Vtab_Size;
extern char *TextModeFont;
extern char *TeXModeFont;

extern int *dec_buf_ptr();
```

---

† Department of Mathematics, Josai University, Keyakidai, Sakado, Saitama, 350-02, JAPAN (kamano@po.iijnet.or.jp)

‡ Department of Mathematics, Josai University, Keyakidai, Sakado, Saitama, 350-02, JAPAN (m93093@crescent.josai.ac.jp)

```
extern int *inc_buf_ptr();
extern int *fgetc2buffer();
extern int *get_phrase();
extern int search_line();
extern int get_comment_flag();
extern int get_tex_flag();
extern int str_cmp();
extern int parse_options();

extern void merge_ntt_ascii();
extern flag_char *get_flag_char();
extern void fprintf_documentstyle();
extern void fprintf_footline();
extern void input_user_style();
extern int choose_tt_font();
extern void fprintf_line_number();
```

/* ——————————————— Text to TEX translator ——————————————— */

/* `Text2tex()` simply translates text data into TEX data. The most important parts of this function are font selection and mode changes. Typewriter font is used in text mode and roman font is chosen otherwise. One of three modes, text mode, quasi-TEX mode and TEX mode, is selected according to

`get_flag_char(fptr)->flag`=0, 1, 2

respectively. A character `get_flag_char(fptr)->character` is translated into an approprite sequence of characters. Especially, in TEX mode, every character got from input file is passed to output file transparently.
*/

```
void text2tex(cptr, fptr)
char *cptr[];
FILE *fptr[];
{
    int char_counter = 0;             /* character counter                       */
    long line_counter = 0;            /* line counter                            */
    int page_len = 1;                 /* lines/page counter                      */
    int qt_counter = 0;               /* single quotation counter                */
    int dqt_counter = 0;              /* double quotation counter                */
    int space_counter = 0;            /* space counter                           */
    int prev_flag = 0;                /* previous flag                           */
    int prev_char = 0x20;             /* previous character                      */
    int bf_flag = 0;                  /* bold face flag                          */
    int skip_amount;                  /* skip amount of tabulation               */
    int stat_flag = 0;                /* status flag                             */
    int nl_flag = 0;                  /* new line flag                           */
    int rm_flag = 0;                  /* cmr font flag                           */
    int tt_flag = 0;                  /* cmtt font flag                          */
    int TT_flag = 0;                  /* choose cmtt font in quasi-TEX mode ?    */
    int doc_flag = 0;                 /* document style flag                     */
    int warn_flag1 = 0;               /* circumventing TeX mode warning flag     */
    int warn_flag2 = 0;               /* too long lines warning flag             */

    int i, *cptr1, *cptr2, *cptr3, *cptr4, *cptr5, *cptr6, *cptr7;
    flag_char *ptr;
```

```
    while (((ptr = get_flag_char(fptr))->character) != EOF)
      {

/* ——————————————— opening message ——————————————— */
/* Here src2tex [resp. src2latex] output preliminary statement of TEX        */

        if (stat_flag == 0)
          {
            ++stat_flag;
            /* set TT_flag at the beginning of translation                    */
            TT_flag = choose_tt_font(ptr->buffer);
#ifndef DEBUGGING
            /* translating message                                            */
#ifdef LATEX
            fprintf(stderr, "src2latex: translating ...");
#else
            fprintf(stderr, "src2tex: translating ...");
#endif
#endif
#ifndef PLAIN
#ifdef LATEX
            /* LaTeX documentstyle                                            */
            fprintf_documentstyle(ptr->buffer, fptr);
#else
            /* TeX footline                                                   */
            fprintf_footline(cptr, fptr);
#endif
            /* baselineskip amount */
            if (Page_Len_Max >= 0)
              {
                fprintf(fptr[1], "\n");
                fprintf(fptr[1], "\\baselineskip=0pt\n");
              }
#endif
            /* input user's style file                                        */
            fprintf(fptr[1], "\n");
            input_user_style(fptr);

            /* absorb differences between ASCII and NTT JTeX                  */
#ifdef ASCII
#ifdef NTT
            /* NTT+ASCII                                                      */
            fprintf(fptr[1], "\n");
            merge_ntt_ascii(fptr);
#endif
#else
#ifdef NTT
            /* NTT                                                            */
            fprintf(fptr[1], "\n");
            merge_ntt_ascii(fptr);
#endif
#endif
```

```
            fprintf(fptr[1], "\n");
            fprintf(fptr[1], "%s\n", TextModeFont);
            fprintf(fptr[1], "\n");
            if (Page_Len_Max < 0)
              fprintf(fptr[1], "\\noindent\n");
            else
              {
#ifdef LATEX
                fprintf(fptr[1], "\\hfill");
                fprintf_line_number(fptr, line_counter);
#else
                fprintf(fptr[1], "\\hfill\n\n\\item{\\tt %d:\\ }\n",
                        line_counter + 1);
#endif
              }
#ifdef LATEX
            /* If there exists a string "{\null}" then skip it.                          */
            if ((char_counter == 0) && (doc_flag == 0))
              {
                if (BAS_flag != 0)
                  if (search_line(ptr->buffer, "'{\null}")
                      || search_line(ptr->buffer, "1'{\null}")
                      || search_line(ptr->buffer, "10'{\null}")
                      || search_line(ptr->buffer, "100'{\null}")
                      || search_line(ptr->buffer, "1000'{\null}")
                      || search_line(ptr->buffer, "rem{\null}")
                      || search_line(ptr->buffer, "1rem{\null}")
                      || search_line(ptr->buffer, "10rem{\null}")
                      || search_line(ptr->buffer, "100rem{\null}")
                      || search_line(ptr->buffer, "1000rem{\null}"))
                    nl_flag = 1;
                if (C_flag != 0)
                  if (search_line(ptr->buffer, "/*{\\null}*/")
                      || search_line(ptr->buffer, "//{\\null}"))
                    nl_flag = 1;
                if (CBL_flag != 0)
                  if (search_line(ptr->buffer, "*{\\null}")
                      || search_line(ptr->buffer, "/{\\null}"))
                    nl_flag = 1;
                if (F77_flag != 0)
                  if (search_line(ptr->buffer, "c{\\null}")
                      || search_line(ptr->buffer, "*{\\null}"))
                    nl_flag = 1;
                if (LISP_flag != 0)
                  {
                    if (search_line(ptr->buffer, ";{\\null}"))
                      nl_flag = 1;
                    if (search_line(ptr->buffer, ";;{\\null}"))
                      nl_flag = 1;
                    if (search_line(ptr->buffer, ";;;{\\null}"))
                      nl_flag = 1;
                    if (search_line(ptr->buffer, ";;;;{\\null}"))
                      nl_flag = 1;
```

```
                if (search_line(ptr->buffer, ";;;;;{\\null}"))
                  nl_flag = 1;
              }
            if (MAKE_flag != 0)
              if (search_line(ptr->buffer, "#{\\null}"))
                nl_flag = 1;
            if (PAS_flag != 0)
              if (search_line(ptr->buffer, "{{\\null}}")
                  || search_line(ptr->buffer, "(*{\\null}*)"))
                nl_flag = 1;
            if (PERL_flag != 0)
              if (search_line(ptr->buffer, "#{\\null}"))
                nl_flag = 1;
            if (SH_flag != 0)
              if (search_line(ptr->buffer, "#{\\null}"))
                nl_flag = 1;
            if (TCL_flag != 0)
              if (search_line(ptr->buffer, "#{\\null}"))
                nl_flag = 1;
            if (MAP_flag != 0)
              if (search_line(ptr->buffer, "#{\\null}"))
                nl_flag = 1;
            if (MAT_flag != 0)
              if (search_line(ptr->buffer, "(*{\\null}*)"))
                nl_flag = 1;
            if (MLAB_flag != 0)
              if (search_line(ptr->buffer, "#{\\null}")
                  || search_line(ptr->buffer, "%{\\null}"))
                nl_flag = 1;
            if (RED_flag != 0)
              if (search_line(ptr->buffer, "%{\\null}")
                  || search_line(ptr->buffer, "COMMENT{\\null};")
                  || search_line(ptr->buffer, "comment{\\null};"))
                nl_flag = 1;
            if (nl_flag != 0)
              {
                ++doc_flag;
                while ((ptr->character != EOF)
                       && ((char)ptr->character != '\n'))
                  ptr = get_flag_char(fptr);
                continue;
              }
          }
#endif
        }
      if ((char_counter == 0) && (line_counter <= 3)
          && (doc_flag == 0))
        {
          if (CBL_flag != 0)
            {
              if (search_line(ptr->buffer, "*{\\null}")
                  || search_line(ptr->buffer, "/{\\null}"))
                nl_flag = 1;
```

```
        if (search_line(ptr->buffer, "000001*{\\null}")
            || search_line(ptr->buffer, "000001/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000002*{\\null}")
            || search_line(ptr->buffer, "000002/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000003*{\\null}")
            || search_line(ptr->buffer, "000003/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000004*{\\null}")
            || search_line(ptr->buffer, "000004/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000010*{\\null}")
            || search_line(ptr->buffer, "000010/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000020*{\\null}")
            || search_line(ptr->buffer, "000020/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000030*{\\null}")
            || search_line(ptr->buffer, "000030/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000040*{\\null}")
            || search_line(ptr->buffer, "000040/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000100*{\\null}")
            || search_line(ptr->buffer, "000100/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000200*{\\null}")
            || search_line(ptr->buffer, "000200/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000300*{\\null}")
            || search_line(ptr->buffer, "000300/{\\null}"))
          nl_flag = 1;
        if (search_line(ptr->buffer, "000400*{\\null}")
            || search_line(ptr->buffer, "000400/{\\null}"))
          nl_flag = 1;
      }
    if (nl_flag != 0)
      {
        ++doc_flag;
        while ((ptr->character != EOF)
               && ((char)ptr->character != '\n'))
          ptr = get_flag_char(fptr);
        continue;
      }
  }
```

/* If there exits a COBOL comment sign / , **src2tex** output page eject code. */

```
if ((ptr->flag == 1)
    && (CBL_flag != 0) && (char_counter == 0))
  {
    cptr1 = ptr->buffer;
    cptr2 = inc_buf_ptr(cptr1);
    cptr3 = inc_buf_ptr(cptr2);
```

```
          cptr4 = inc_buf_ptr(cptr3);
          cptr5 = inc_buf_ptr(cptr4);
          cptr6 = inc_buf_ptr(cptr5);
          cptr7 = inc_buf_ptr(cptr6);
          if (*cptr7 == '/')
            fprintf(fptr[1], "\\vfill\\eject\n\n\\noindent\n");
        }
      if ((prev_flag > 1) && (ptr->flag == 0))
        {
          fprintf(stderr,
                  "\nError: unexpected end of TeX-mode in %s\n", cptr[0]);
          fprintf(stderr,
                  "       illegal transition TeX-mode -> Text-mode\n");
          exit(EXIT_FAILURE);
        }

/* ______________________________ font selection ______________________________ */

      if ((prev_flag == 0) && (ptr->flag > 0))
        {
#ifndef DEBUGGING
          /* i-am-working message                                                */
          fprintf(stderr, ".");
#endif
          /* We usually select TeXModeFont  as follows.                          */
          if (((C_flag == 0) && (F77_flag == 0)
               && (MAKE_flag == 0) && (PAS_flag == 0)
               && (PERL_flag == 0) && (SH_flag == 0) && (TCL_flag == 0)
               && (MAP_flag == 0) && (MAT_flag == 0) && (MLAB_flag == 0))
              || (char_counter >= Htab_Size))
            {
              if (TT_flag != 0)
                fprintf(fptr[1], "%s", TextModeFont);
              else
                fprintf(fptr[1], "%s", TeXModeFont);
            }

          /* If there exists a C comment area consists of several quasi-TeXmode lines, */
          /* src2tex uses cmtt font instead of cmr font                          */
          if ((C_flag != 0)
              && (char_counter < Htab_Size))
            {
              tt_flag = 0;
              cptr1 = ptr->buffer;
              cptr2 = inc_buf_ptr(cptr1);
              cptr3 = inc_buf_ptr(cptr2);
              cptr4 = inc_buf_ptr(cptr3);
              cptr5 = inc_buf_ptr(cptr4);
              cptr6 = inc_buf_ptr(cptr5);
              for (i = 0; i < 1024; ++i)
                {
                  if ((*cptr1 == '{') && (*cptr2 == '\\'))
                    {
```

```
                rm_flag = 1;
                tt_flag = 0;
                break;
              }
#ifdef LATEX
            if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
            if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
              {
                rm_flag = 1;
                tt_flag = 0;
                break;
              }
#ifdef LATEX
            if ((*cptr1 == '\\') && (*cptr2 == '('))
#else
            if ((*cptr1 != '\\') && (*cptr2 == '$'))
#endif
              {
                rm_flag = 1;
                tt_flag = 0;
                break;
              }
            if (((*cptr1 == '*') && (*cptr2 == '/'))
                && ((*cptr3 == '\n') || (*cptr3 == '\r'))
                && ((*cptr4 == '/') && (*cptr5 == '*')))
              {
                rm_flag = 0;
                tt_flag = 2;
                break;
              }
            if (((*cptr1 == '*') && (*cptr2 == '/'))
                && ((*cptr3 == '\n') || (*cptr3 == '\r'))
                && ((*cptr4 == '\n') || (*cptr4 == '\r'))
                && ((*cptr5 == '/') && (*cptr6 == '*')))
              {
                rm_flag = 0;
                tt_flag = 2;
                break;
              }
            if (((*cptr1 == '*') && (*cptr2 == '/'))
                && ((*cptr3 == '\n') || (*cptr3 == '\r')))
              {
                rm_flag = 0;
                tt_flag = 1;
                break;
              }
            if (((*cptr1 == '\n') || (*cptr1 == '\r'))
                && (*cptr2 == '/') && (*cptr3 == '*'))
              {
                rm_flag = 0;
                tt_flag =1;
```

```
            break;
           }
          if ((*cptr1 == '\n') || (*cptr1 == '\r'))
           tt_flag = 1;
          cptr1 = inc_buf_ptr(cptr1);
          cptr2 = inc_buf_ptr(cptr2);
          cptr3 = inc_buf_ptr(cptr3);
          cptr4 = inc_buf_ptr(cptr4);
          cptr5 = inc_buf_ptr(cptr5);
          cptr6 = inc_buf_ptr(cptr6);
         }
#ifdef DEBUGGING
        if (tt_flag != 0)
         printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
#endif
        if ((tt_flag > 1) && (warn_flag1 <= 5))
         {
          ++warn_flag1;
#ifdef DEBUGGING
          printf("text2tex(): warn_flag1 is set to %d\n",
              warn_flag1);
#endif
          if (warn_flag1 > 5)
           {
            if (warn_flag2 <= 2)
             fprintf(stderr, "\n");
            fprintf(stderr,
              "Warning: It is better to use TeX-mode\n");
            fprintf(stderr,
              "    when you write long comment in C.\n");
           }
         }
        if ((TT_flag == 1)
          || ((rm_flag == 0) && (tt_flag != 0)))
         fprintf(fptr[1], "%s", TextModeFont);
        else
         fprintf(fptr[1], "%s", TeXModeFont);
       }

      /* If there exists a FORTRAN comment area consists of several quasi-TeX  */
      /* mode lines, src2tex uses cmtt font instead of cmr font                 */
      if ((F77_flag != 0)
        && (char_counter < Htab_Size))
       {
        tt_flag = 0;
        cptr1 = ptr->buffer;
        cptr2 = inc_buf_ptr(cptr1);
        cptr3 = inc_buf_ptr(cptr2);
        for (i = 0; i < 1024; ++i)
         {
          if ((*cptr1 == '{') && (*cptr2 == '\\'))
           {
            rm_flag = 1;
```

```
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
                    if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '('))
#else
                    if ((*cptr1 != '\\') && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
                    if (((*cptr1 != '\n') && (*cptr1 != '\r'))
                        && ((*cptr2 == '*') || (*cptr2 == 'C')
                            || (*cptr2 == 'c'))
                        && ((*cptr3 == '\n') || (*cptr3 == '\r')))
                      {
                        rm_flag = 0;
                        tt_flag = 2;
                        break;
                      }
                    if ((*cptr1 == '\n') || (*cptr1 == '\r'))
                      tt_flag = 1;
                    cptr1 = inc_buf_ptr(cptr1);
                    cptr2 = inc_buf_ptr(cptr2);
                    cptr3 = inc_buf_ptr(cptr3);
                  }
#ifdef DEBUGGING
                if (tt_flag != 0)
                  printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
#endif
                if ((tt_flag > 1) && (warn_flag1 <= 5))
                  {
                    ++warn_flag1;
#ifdef DEBUGGING
                    printf(
                        "text2tex(): warn_flag1 is set to %d\n", warn_flag1);
#endif
                    if (warn_flag1 > 5)
                      {
                        if (warn_flag2 <= 2)
                          fprintf(stderr, "\n");
```

```
                    fprintf(stderr,
                    "Warning: It is better to use TeX-mode\n");
                    fprintf(stderr,
                    "         when you write long comment in FORTRAN.\n");
                  }
              }
            if ((TT_flag == 1)
                || ((rm_flag == 0) && (tt_flag != 0)))
              fprintf(fptr[1], "%s", TextModeFont);
            else
              fprintf(fptr[1], "%s", TeXModeFont);
          }

        /* If there exists a MAKE comment area consists of several quasi-TeX */
        /* mode lines, src2tex uses cmtt font instead of cmr font            */
        if ((MAKE_flag != 0)
            && (char_counter < Htab_Size))
          {
            tt_flag = 0;
            cptr1 = ptr->buffer;
            cptr2 = inc_buf_ptr(cptr1);
            cptr3 = inc_buf_ptr(cptr2);
            for (i = 0; i < 1024; ++i)
              {
                if ((*cptr1 == '{') && (*cptr2 == '\\'))
                  {
                    rm_flag = 1;
                    tt_flag = 0;
                    break;
                  }
#ifdef LATEX
                if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
                if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
                  {
                    rm_flag = 1;
                    tt_flag = 0;
                    break;
                  }
#ifdef LATEX
                if ((*cptr1 == '\\') && (*cptr2 == '('))
#else
                if (((*cptr1 == '\t') || (*cptr1 == ' '))
                    && (*cptr2 == '$') && (*cptr3 != '('))
#endif
                  {
                    rm_flag = 1;
                    tt_flag = 0;
                    break;
                  }
                if (((*cptr1 != '\n') && (*cptr1 != '\r'))
                    && (*cptr2 == '#')
```

```
                    && ((*cptr3 == '\n') || (*cptr3 == '\r')))
                  {
                    rm_flag = 0;
                    tt_flag = 2;
                    break;
                  }
                if ((*cptr1 == '\n') || (*cptr1 == '\r'))
                  tt_flag = 1;
                cptr1 = inc_buf_ptr(cptr1);
                cptr2 = inc_buf_ptr(cptr2);
                cptr3 = inc_buf_ptr(cptr3);
              }
#ifdef DEBUGGING
            if (tt_flag != 0)
              printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
#endif
            if ((tt_flag > 1) && (warn_flag1 <= 5))
              {
                ++warn_flag1;
#ifdef DEBUGGING
                printf("text2tex(): warn_flag1 is set to %d\n",
                       warn_flag1);
#endif
                if (warn_flag1 > 5)
                  {
                    if (warn_flag2 <= 2)
                      fprintf(stderr, "\n");
                    fprintf(stderr,
                    "Warning: It is better to use TeX-mode\n");
                    fprintf(stderr,
                    "          when you write long comment in MAKE.\n");
                  }
              }
            if ((TT_flag == 1)
                || ((rm_flag == 0) && (tt_flag != 0)))
              fprintf(fptr[1], "%s", TextModeFont);
            else
              fprintf(fptr[1], "%s", TeXModeFont);
          }

        /* If there exists a PASCAL comment area consists of several quasi-TEX */
        /* mode lines, src2tex uses cmtt font instead of cmr font */
        if ((PAS_flag != 0)
            && (char_counter < Htab_Size))
          {
            tt_flag = 0;
            cptr1 = ptr->buffer;
            cptr2 = inc_buf_ptr(cptr1);
            cptr3 = inc_buf_ptr(cptr2);
            cptr4 = inc_buf_ptr(cptr3);
            cptr5 = inc_buf_ptr(cptr4);
            cptr6 = inc_buf_ptr(cptr5);
            for (i = 0; i < 1024; ++i)
```

```
                    {
                      if ((*cptr1 == '{') && (*cptr2 == '\\'))
                        {
                          rm_flag = 1;
                          tt_flag = 0;
                          break;
                        }
#ifdef LATEX
                      if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
                      if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
                        {
                          rm_flag = 1;
                          tt_flag = 0;
                          break;
                        }
#ifdef LATEX
                      if ((*cptr1 == '\\') && (*cptr2 == '('))
#else
                      if ((*cptr1 != '\\') && (*cptr2 == '$'))
#endif
                        {
                          rm_flag = 1;
                          tt_flag = 0;
                          break;
                        }
                      if ((*cptr1 == '}')
                          && ((*cptr2 == '\n') || (*cptr2 == '\r'))
                          && (*cptr3 == '{'))
                        {
                          rm_flag = 0;
                          tt_flag = 2;
                          break;
                        }
                      if (((*cptr1 == '*') && (*cptr2 == ')'))
                          && ((*cptr3 == '\n') || (*cptr3 == '\r'))
                          && ((*cptr4 == '(') && (*cptr5 == '*')))
                        {
                          rm_flag = 0;
                          tt_flag = 2;
                          break;
                        }
                      if ((*cptr1 == '}')
                          && ((*cptr2 == '\n') || (*cptr2 == '\r'))
                          && ((*cptr3 == '\n') || (*cptr3 == '\r'))
                          && (*cptr4 == '{'))
                        {
                          rm_flag = 0;
                          tt_flag = 2;
                          break;
                        }
                      if (((*cptr1 == '*') && (*cptr2 == ')'))
```

```
                      && ((*cptr3 == '\n') || (*cptr3 == '\r'))
                      && ((*cptr4 == '\n') || (*cptr4 == '\r'))
                      && ((*cptr5 == '(') && (*cptr6 == '*')))
                    {
                      rm_flag = 0;
                      tt_flag = 2;
                      break;
                    }
                  if ((*cptr1 == '}')
                      && ((*cptr2 == '\n') || (*cptr2 == '\r')))
                    {
                      rm_flag = 0;
                      tt_flag = 1;
                      break;
                    }
                  if (((*cptr1 == '*') && (*cptr2 == ')'))
                      && ((*cptr3 == '\n') || (*cptr3 == '\r')))
                    {
                      rm_flag = 0;
                      tt_flag = 1;
                      break;
                    }
                  if (((*cptr1 == '\n') || (*cptr1 == '\r'))
                      && (*cptr2 == '}'))
                    {
                      rm_flag = 0;
                      tt_flag = 1;
                      break;
                    }
                  if (((*cptr1 == '\n') || (*cptr1 == '\r'))
                      && (*cptr2 == '(') && (*cptr3 == '*'))
                    {
                      rm_flag = 0;
                      tt_flag = 1;
                      break;
                    }
                  if ((*cptr1 == '\n') || (*cptr1 == '\r'))
                    tt_flag = 1;
                  cptr1 = inc_buf_ptr(cptr1);
                  cptr2 = inc_buf_ptr(cptr2);
                  cptr3 = inc_buf_ptr(cptr3);
                  cptr4 = inc_buf_ptr(cptr4);
                  cptr5 = inc_buf_ptr(cptr5);
                  cptr6 = inc_buf_ptr(cptr6);
                }
#ifdef DEBUGGING
              if (tt_flag != 0)
                printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
#endif
              if ((tt_flag > 1) && (warn_flag1 <= 5))
                {
                  ++warn_flag1;
#ifdef DEBUGGING
```

```
                    printf("text2tex(): warn_flag1 is set to %d\n",
                           warn_flag1);
#endif
                    if (warn_flag1 > 5)
                      {
                        if (warn_flag2 <= 2)
                          fprintf(stderr, "\n");
                        fprintf(stderr,
                        "Warning: It is better to use TeX-mode\n");
                        fprintf(stderr,
                        "          when you write long comment in PASCAL.\n");
                      }
                  }
                if ((TT_flag == 1)
                    || ((rm_flag == 0) && (tt_flag != 0)))
                  fprintf(fptr[1], "%s", TextModeFont);
                else
                  fprintf(fptr[1], "%s", TeXModeFont);
              }

            /* If there exists a PERL comment area consists of several quasi-TeX    */
            /* mode lines, src2tex uses cmtt font instead of cmr font                */
            if ((PERL_flag != 0)
                && (char_counter < Htab_Size))
              {
                tt_flag = 0;
                cptr1 = ptr->buffer;
                cptr2 = inc_buf_ptr(cptr1);
                cptr3 = inc_buf_ptr(cptr2);
                for (i = 0; i < 1024; ++i)
                  {
                    if ((*cptr1 == '{') && (*cptr2 == '\\'))
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
                    if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '('))
#else
                    if (((*cptr1 == '\t') || (*cptr1 == ' '))
                        && (*cptr2 == '$') && (*cptr3 == '\\'))
#endif
```

```
                {
                  rm_flag = 1;
                  tt_flag = 0;
                  break;
                }
              if (((*cptr1 != '\n') && (*cptr1 != '\r'))
                  && (*cptr2 == '#')
                  && ((*cptr3 == '\n') || (*cptr3 == '\r')))
                {
                  rm_flag = 0;
                  tt_flag = 2;
                  break;
                }
              if ((*cptr1 == '\n') || (*cptr1 == '\r'))
                tt_flag = 1;
              cptr1 = inc_buf_ptr(cptr1);
              cptr2 = inc_buf_ptr(cptr2);
              cptr3 = inc_buf_ptr(cptr3);
            }
#ifdef DEBUGGING
          if (tt_flag != 0)
            {
              printf ("text2tex(): rm_flag is set to %d\n", rm_flag);
              printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
            }
#endif
          if ((tt_flag > 1) && (warn_flag1 <= 5))
            {
              ++warn_flag1;
#ifdef DEBUGGING
              printf("text2tex(): warn_flag1 is set to %d\n",
                     warn_flag1);
#endif
              if (warn_flag1 > 5)
                {
                  if (warn_flag2 <= 2)
                    fprintf(stderr, "\n");
                  fprintf(stderr,
                  "Warning: It is better to use TeX-mode\n");
                  fprintf(stderr,
                  "          when you write long comment in PERL.\n");
                }
            }
          if ((TT_flag == 1)
              || ((rm_flag == 0) && (tt_flag != 0)))
            fprintf(fptr[1], "%s", TextModeFont);
          else
            fprintf(fptr[1], "%s", TeXModeFont);
        }
```

/* If there exists a SHELL comment area consists of several quasi-TEX */
/* mode lines, **src2tex** uses cmtt font instead of cmr font */

```
      if ((SH_flag != 0)
```

```
                && (char_counter < Htab_Size))
              {
                tt_flag = 0;
                cptr1 = ptr->buffer;
                cptr2 = inc_buf_ptr(cptr1);
                cptr3 = inc_buf_ptr(cptr2);
                for (i = 0; i < 1024; ++i)
                  {
                    if ((*cptr1 == '{') && (*cptr2 == '\\'))
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
                    if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '('))
#else
                    if (((*cptr1 == '\t') || (*cptr1 == ' '))
                        && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
                    if (((*cptr1 != '\n') && (*cptr1 != '\r'))
                        && (*cptr2 == '#')
                        && ((*cptr3 == '\n') || (*cptr3 == '\r')))
                      {
                        rm_flag = 0;
                        tt_flag = 2;
                        break;
                      }
                    if ((*cptr1 == '\n') || (*cptr1 == '\r'))
                      tt_flag = 1;
                    cptr1 = inc_buf_ptr(cptr1);
                    cptr2 = inc_buf_ptr(cptr2);
                    cptr3 = inc_buf_ptr(cptr3);
                  }
#ifdef DEBUGGING
                if (tt_flag != 0)
                  printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
#endif
```

```
                if ((tt_flag > 1) && (warn_flag1 <= 5))
                  {
                    ++warn_flag1;
#ifdef DEBUGGING
                    printf("text2tex(): warn_flag1 is set to %d\n",
                           warn_flag1);
#endif
                    if (warn_flag1 > 5)
                      {
                        if (warn_flag2 <= 2)
                          fprintf(stderr, "\n");
                        fprintf(stderr,
                        "Warning: It is better to use TeX-mode\n");
                        fprintf(stderr,
                        "          when you write long comment in SHELL.\n");
                      }
                  }
                if ((TT_flag == 1)
                    || ((rm_flag == 0) && (tt_flag != 0)))
                  fprintf(fptr[1], "%s", TextModeFont);
                else
                  fprintf(fptr[1], "%s", TeXModeFont);
              }

            /* If there exists a TCL/TK comment area consists of several quasi-TeX   */
            /* mode lines, src2tex uses cmtt font instead of cmr font                */
            if ((TCL_flag != 0)
                && (char_counter < Htab_Size))
              {
                tt_flag = 0;
                cptr1 = ptr->buffer;
                cptr2 = inc_buf_ptr(cptr1);
                cptr3 = inc_buf_ptr(cptr2);
                for (i = 0; i < 1024; ++i)
                  {
                    if ((*cptr1 == '{') && (*cptr2 == '\\'))
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
                    if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '('))
```

```
#else
                    if (((*cptr1 == '\t') || (*cptr1 == ' '))
                        && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
                    if (((*cptr1 != '\n') && (*cptr1 != '\r'))
                        && (*cptr2 == '#')
                        && ((*cptr3 == '\n') || (*cptr3 == '\r')))
                      {
                        rm_flag = 0;
                        tt_flag = 2;
                        break;
                      }
                    if ((*cptr1 == '\n') || (*cptr1 == '\r'))
                      tt_flag = 1;
                    cptr1 = inc_buf_ptr(cptr1);
                    cptr2 = inc_buf_ptr(cptr2);
                    cptr3 = inc_buf_ptr(cptr3);
                  }
#ifdef DEBUGGING
                if (tt_flag != 0)
                  printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
#endif
                if ((tt_flag > 1) && (warn_flag1 <= 5))
                  {
                    ++warn_flag1;
#ifdef DEBUGGING
                    printf("text2tex(): warn_flag1 is set to %d\n",
                           warn_flag1);
#endif
                    if (warn_flag1 > 5)
                      {
                        if (warn_flag2 <= 2)
                          fprintf(stderr, "\n");
                        fprintf(stderr,
                        "Warning: It is better to use TeX-mode\n");
                        fprintf(stderr,
                        "          when you write long comment in TCL/TK.\n");
                      }
                  }
                if ((TT_flag == 1)
                    || ((rm_flag == 0) && (tt_flag != 0)))
                  fprintf(fptr[1], "%s", TextModeFont);
                else
                  fprintf(fptr[1], "%s", TeXModeFont);
              }
```

/* If there exists a MAPLE comment area consists of several quasi-TEX */
/* mode lines, **src2tex** uses cmtt font instead of cmr font */

```
            if ((MAP_flag != 0)
                && (char_counter < Htab_Size))
              {
                tt_flag = 0;
                cptr1 = ptr->buffer;
                cptr2 = inc_buf_ptr(cptr1);
                cptr3 = inc_buf_ptr(cptr2);
                for (i = 0; i < 1024; ++i)
                  {
                    if ((*cptr1 == '{') && (*cptr2 == '\\'))
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
                    if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '('))
#else
                    if ((*cptr1 != '\\') && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
                    if (((*cptr1 != '\n') && (*cptr1 != '\r'))
                        && (*cptr2 == '#')
                        && ((*cptr3 == '\n') || (*cptr3 == '\r')))
                      {
                        rm_flag = 0;
                        tt_flag = 2;
                        break;
                      }
                    if ((*cptr1 == '\n') || (*cptr1 == '\r'))
                      tt_flag = 1;
                    cptr1 = inc_buf_ptr(cptr1);
                    cptr2 = inc_buf_ptr(cptr2);
                    cptr3 = inc_buf_ptr(cptr3);
                  }
#ifdef DEBUGGING
                if (tt_flag != 0)
                  printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
#endif
```

```
                if ((tt_flag > 1) && (warn_flag1 <= 5))
                  {
                    ++warn_flag1;
#ifdef DEBUGGING
                    printf("text2tex(): warn_flag1 is set to %d\n",
                           warn_flag1);
#endif
                    if (warn_flag1 > 5)
                      {
                        if (warn_flag2 <= 2)
                          fprintf(stderr, "\n");
                        fprintf(stderr,
                        "Warning: It is better to use TeX-mode\n");
                        fprintf(stderr,
                        "         when you write long comment in MAPLE.\n");
                      }
                  }
                if ((TT_flag == 11)
                    || ((rm_flag == 0) && (tt_flag != 0)))
                  fprintf(fptr[1], "%s", TextModeFont);
                else
                  fprintf(fptr[1], "%s", TeXModeFont);
              }

            /* If there exists a MATHEMATICA comment area consists of several quasi-TeX      */
            /* mode lines, src2tex uses cmtt font instead of cmr font                       */
            if ((MAT_flag != 0)
                && (char_counter < Htab_Size))
              {
                tt_flag = 0;
                cptr1 = ptr->buffer;
                cptr2 = inc_buf_ptr(cptr1);
                cptr3 = inc_buf_ptr(cptr2);
                cptr4 = inc_buf_ptr(cptr3);
                cptr5 = inc_buf_ptr(cptr4);
                cptr6 = inc_buf_ptr(cptr5);
                for (i = 0; i < 1024; ++i)
                  {
                    if ((*cptr1 == '{') && (*cptr2 == '\\'))
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
                    if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
```

```
                          }
#ifdef LATEX
                        if ((*cptr1 == '\\') && (*cptr2 == '('))
#else
                        if ((*cptr1 != '\\') && (*cptr2 == '$'))
#endif
                          {
                            rm_flag = 1;
                            tt_flag = 0;
                            break;
                          }
                        if (((*cptr1 == '*') && (*cptr2 == ')'))
                            && ((*cptr3 == '\n') || (*cptr3 == '\r'))
                            && ((*cptr4 == '(') && (*cptr5 == '*')))
                          {
                            rm_flag = 0;
                            tt_flag = 2;
                            break;
                          }
                        if (((*cptr1 == '*') && (*cptr2 == ')'))
                            && ((*cptr3 == '\n') || (*cptr3 == '\r'))
                            && ((*cptr4 == '\n') || (*cptr4 == '\r'))
                            && ((*cptr5 == '(') && (*cptr6 == '*')))
                          {
                            rm_flag = 0;
                            tt_flag = 2;
                            break;
                          }
                        if (((*cptr1 == '*') && (*cptr2 == ')'))
                            && ((*cptr3 == '\n') || (*cptr3 == '\r')))
                          {
                            rm_flag = 0;
                            tt_flag = 1;
                            break;
                          }
                        if (((*cptr1 == '\n') || (*cptr1 == '\r'))
                            && (*cptr2 == '(') && (*cptr3 == '*'))
                          {
                            rm_flag = 0;
                            tt_flag = 1;
                            break;
                          }
                        if ((*cptr1 == '\n') || (*cptr1 == '\r'))
                          tt_flag = 1;
                        cptr1 = inc_buf_ptr(cptr1);
                        cptr2 = inc_buf_ptr(cptr2);
                        cptr3 = inc_buf_ptr(cptr3);
                        cptr4 = inc_buf_ptr(cptr4);
                        cptr5 = inc_buf_ptr(cptr5);
                        cptr6 = inc_buf_ptr(cptr6);
                      }
#ifdef DEBUGGING
                    if ((rm_flag == 0) && (tt_flag != 0))
```

```
                 printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
#endif
               if ((tt_flag > 1) && (warn_flag1 <= 5))
                 {
                   ++warn_flag1;
#ifdef DEBUGGING
                   printf("text2tex(): warn_flag1 is set to %d\n",
                          warn_flag1);
#endif
                   if (warn_flag1 > 5)
                     {
                       if (warn_flag2 <= 2)
                         fprintf(stderr, "\n");
                       fprintf(stderr,
                               "Warning: It is better to use TeX-mode\n");
                       fprintf(stderr,
                               "          when you write long comment in");
                       fprintf(stderr,
                               " MATHEMATICA.\n");
                     }
                 }
               if ((TT_flag == 1)
                   || ((rm_flag == 0) && (tt_flag != 0)))
                 fprintf(fptr[1], "%s", TextModeFont);
               else
                 fprintf(fptr[1], "%s", TeXModeFont);
             }

            /* If there exists a MATLAB comment area consists of several quasi-TeX     */
            /* mode lines, src2tex uses cmtt font instead of cmr font                  */
            if ((MLAB_flag != 0)
                && (char_counter < Htab_Size))
              {
                tt_flag = 0;
                cptr1 = ptr->buffer;
                cptr2 = inc_buf_ptr(cptr1);
                cptr3 = inc_buf_ptr(cptr2);
                for (i = 0; i < 1024; ++i)
                  {
                    if ((*cptr1 == '{') && (*cptr2 == '\\'))
                      {
                        rm_flag = 1;
                        tt_flag = 0;
                        break;
                      }
#ifdef LATEX
                    if ((*cptr1 == '\\') && (*cptr2 == '['))
#else
                    if ((*cptr1 == '$') && (*cptr2 == '$'))
#endif
                      {
                        rm_flag = 1;
                        tt_flag = 0;
```

```
                      break;
                    }
#ifdef LATEX
                  if ((*cptr1 == '\\') && (*cptr2 == '('))
#else
                  if ((*cptr1 != '\\') && (*cptr2 == '$'))
#endif
                    {
                      rm_flag = 1;
                      tt_flag = 0;
                      break;
                    }
                  if ((*cptr1 == '\n')
                      && ((*cptr2 == '#') || (*cptr2 == '%'))
                      && ((*cptr3 == '\t') || (*cptr3 == ' ')))
                    {
                      rm_flag = 0;
                      tt_flag = 2;
                      break;
                    }
                  if (((*cptr1 != '\n') && (*cptr1 != '\r'))
                      && ((*cptr2 == '#') || (*cptr2 == '%'))
                      && ((*cptr3 == '\n') || (*cptr3 == '\r')))
                    {
                      rm_flag = 0;
                      tt_flag = 2;
                      break;
                    }
                  if ((*cptr1 == '\n') || (*cptr1 == '\r'))
                    tt_flag = 1;
                  cptr1 = inc_buf_ptr(cptr1);
                  cptr2 = inc_buf_ptr(cptr2);
                  cptr3 = inc_buf_ptr(cptr3);
                }
#ifdef DEBUGGING
              if (tt_flag != 0)
                printf ("text2tex(): tt_flag is set to %d\n", tt_flag);
#endif
              if ((tt_flag > 1) && (warn_flag1 <= 5))
                {
                  ++warn_flag1;
#ifdef DEBUGGING
                  printf("text2tex(): warn_flag1 is set to %d\n",
                         warn_flag1);
#endif
                  if (warn_flag1 > 5)
                    {
                      if (warn_flag2 <= 2)
                        fprintf(stderr, "\n");
                      fprintf(stderr,
                              "Warning: It is better to use TeX-mode\n");
                      fprintf(stderr,
                              "          when you write long comment in");
```

```
                    fprintf(stderr,
                            " MATLAB.\n");
                  }
              }
            if ((TT_flag == 1)
                || ((rm_flag == 0) && (tt_flag != 0)))
              fprintf(fptr[1], "%s", TextModeFont);
            else
              fprintf(fptr[1], "%s", TeXModeFont);
          }
      }

    /* choose either TextModeFont or TeXModeFont carefully            */
    if ((prev_flag > 0) && (ptr->flag == 0))
      {
        if(char_counter == 0)
          fprintf(fptr[1], "%s", TextModeFont);
        else
          fprintf(fptr[1], "\n%s", TextModeFont);
      }
    if ((prev_flag > 1) && (ptr->flag == 1))
      {
        if (TT_flag != 0)
          fprintf(fptr[1], "%s", TextModeFont);
        else
          fprintf(fptr[1], "%s", TeXModeFont);
      }
    /* set bold face flag bf_flag for PASCAL and REDUCE                */
    if (ptr->flag == 0)
      if ((prev_char < 'A')
          || ((prev_char > 'Z') && (prev_char < 'a'))
          || (prev_char > 'z'))
        if (((PAS_flag != 0) || (RED_flag != 0))
            && (qt_counter == 0) && (bf_flag == 0))
          {
            bf_flag = get_bf_flag(ptr->buffer);
#ifdef DEBUGGING
            printf("get_bf_flag(): bf_flag = %d\n", bf_flag);
#endif
          }
    /* parsing options                                                 */
    if ((ptr->flag >= 1) && (char_counter == 0))
      {
        if (parse_options(ptr) != 0)
          {
            if (TT_flag != 0)
              fprintf(fptr[1], "%s", TextModeFont);
            else
              fprintf(fptr[1], "%s", TeXModeFont);
          }
      }
    /* commenting out src2tex  escape sequence                         */
    if ((ptr->flag >= 1) && (char_counter == 0))
```

```
          {
            cptr1 = ptr->buffer;
            while (((char)*cptr1 != '\\')
                   && ((char)*cptr1 != '\n') && (*cptr1 != EOF))
              cptr1 = inc_buf_ptr(cptr1);
            if (str_cmp(cptr1, "\\src2tex{") == 0)
              {
                fprintf(fptr[1], "%c ", 0x25);
#ifdef DEBUGGING
                printf("commenting out \\src2tex{ ... }\n");
#endif
              }
          }
        prev_flag = ptr->flag;
        prev_char = ptr->character;
```

/* ―――――――――――――――― text and quasi-TEX mode ―――――――――――――――― */
/* Here **src2tex** [resp. **src2latex**] translates each input character into suitable escape sequence of TEX */

```
        if (ptr->flag <= 1)
          {
            switch (ptr->character)
              {
              case '\0':
                fprintf(fptr[1],
                        "{\\sevenrm N\\kern-.15em\\lower.5ex\\hbox{U}}");
                break;
              case 0x01:
                fprintf(fptr[1],
                        "{\\sevenrm S\\kern-.15em\\lower.5ex\\hbox{H}}");
                break;
              case 0x02:
                fprintf(fptr[1],
                        "{\\sevenrm S\\kern-.15em\\lower.5ex\\hbox{X}}");
                break;
              case 0x03:
                fprintf(fptr[1],
                        "{\\sevenrm E\\kern-.15em\\lower.5ex\\hbox{X}}");
                break;
              case 0x04:
                fprintf(fptr[1],
                        "{\\sevenrm E\\kern-.15em\\lower.5ex\\hbox{T}}");
                break;
              case 0x05:
                fprintf(fptr[1],
                        "{\\sevenrm E\\kern-.15em\\lower.5ex\\hbox{Q}}");
                break;
              case 0x06:
                fprintf(fptr[1],
                        "{\\sevenrm A\\kern-.15em\\lower.5ex\\hbox{K}}");
                break;
              case 0x07:
                fprintf(fptr[1],
```

```
                        "{\\sevenrm B\\kern-.15em\\lower.5ex\\hbox{L}}");
                break;
              case '\b':
                fprintf(fptr[1],
                        "{\\sevenrm B\\kern-.15em\\lower.5ex\\hbox{S}}");
                break;
              case '\t':
                skip_amount = Htab_Size - (char_counter % Htab_Size);
                fprintf(fptr[1], "{%s\\kern%3.3fem}",
                        TextModeFont, (float) SPACE * (float) skip_amount);
                char_counter += skip_amount -1;
                break;
              case '\n':
                if ((Page_Len_Max > 0) && (page_len >= Page_Len_Max))
                  {
                    fprintf(fptr[1], "\n\n\\vfill\\eject\n\n");
                    page_len = 0;
                  }
                ++line_counter;
                ++page_len;
                if (char_counter == 0)
                  {
                    cptr1 = inc_buf_ptr(ptr->buffer);
                    if ((Page_Len_Max < 0) || (*cptr1 == EOF))
                      fprintf(fptr[1], "\\hfill\n\n\\noindent\n");
                    else
                      {
#ifdef LATEX
                        fprintf(fptr[1], "\\hfill");
                        fprintf_line_number(fptr, line_counter);
#else
                        fprintf(fptr[1], "\\hfill\n\n\\item{\\tt %d:\\ }\n",
                                line_counter + 1);
#endif
                      }
                  }
                else
                  {
                    cptr1 = inc_buf_ptr(ptr->buffer);
                    if ((Page_Len_Max < 0) || (*cptr1 == EOF))
                      fprintf(fptr[1], "\n\n\\noindent\n");
                    else
                      {
#ifdef LATEX
                        fprintf_line_number(fptr, line_counter);
#else
                        fprintf(fptr[1], "\n\n\\item{\\tt %d:\\ }\n",
                                line_counter + 1);
#endif
                      }
                  }
                if (ptr->flag == 0)
                  fprintf(fptr[1], "{}");
```

```
                char_counter = -1;
                break;
              case '\v':
                fprintf(fptr[1], "{\\vskip%dex\\relax }", Vtab_Size);
                break;
              case '\f':
/*
                fprintf(fptr[1],
                        " “sevenrm F“kern-.15em“lower.5ex“hboxF ");
*/
                fprintf(fptr[1],
                        "\\vfill\\eject\n\n\\noindent\n");
                break;
              case '\r':
#ifdef UNIX
                fprintf(fptr[1],
                        "{\\sevenrm C\\kern-.15em\\lower.5ex\\hbox{R}}");
#else
                fprintf(fptr[1],
                        "%c", (char) ptr->character);
#endif
                break;
              case 0x0e:
                fprintf(fptr[1],
                        "{\\sevenrm S\\kern-.15em\\lower.5ex\\hbox{O}}");
                break;
              case 0x0f:
                fprintf(fptr[1],
                        "{\\sevenrm S\\kern-.15em\\lower.5ex\\hbox{I}}");
                break;
              case 0x10:
                fprintf(fptr[1],
                        "{\\sevenrm D\\kern-.15em\\lower.5ex\\hbox{L}}");
                break;
              case 0x11:
                fprintf(fptr[1],
                        "{\\sevenrm D\\kern-.15em\\lower.5ex\\hbox{1}}");
                break;
              case 0x12:
                fprintf(fptr[1],
                        "{\\sevenrm D\\kern-.15em\\lower.5ex\\hbox{2}}");
                break;
              case 0x13:
                fprintf(fptr[1],
                        "{\\sevenrm D\\kern-.15em\\lower.5ex\\hbox{3}}");
                break;
              case 0x14:
                fprintf(fptr[1],
                        "{\\sevenrm D\\kern-.15em\\lower.5ex\\hbox{4}}");
                break;
              case 0x15:
                fprintf(fptr[1],
                        "{\\sevenrm N\\kern-.15em\\lower.5ex\\hbox{K}}");
```

```
    break;
  case 0x16:
    fprintf(fptr[1],
            "{\\sevenrm S\\kern-.15em\\lower.5ex\\hbox{Y}}");
    break;
  case 0x17:
    fprintf(fptr[1],
            "{\\sevenrm E\\kern-.15em\\lower.5ex\\hbox{B}}");
    break;
  case 0x18:
    fprintf(fptr[1],
            "{\\sevenrm C\\kern-.15em\\lower.5ex\\hbox{N}}");
    break;
  case 0x19:
    fprintf(fptr[1],
            "{\\sevenrm E\\kern-.15em\\lower.5ex\\hbox{M}}");
    break;
  case 0x1a:
    fprintf(fptr[1],
            "{\\sevenrm S\\kern-.15em\\lower.5ex\\hbox{B}}");
    break;
  case 0x1b:
    fprintf(fptr[1],
            "{\\sevenrm E\\kern-.15em\\lower.5ex\\hbox{C}}");
    break;
  case 0x1c:
    fprintf(fptr[1],
            "{\\sevenrm F\\kern-.15em\\lower.5ex\\hbox{S}}");
    break;
  case 0x1d:
    fprintf(fptr[1],
            "{\\sevenrm G\\kern-.15em\\lower.5ex\\hbox{S}}");
    break;
  case 0x1e:
    fprintf(fptr[1],
            "{\\sevenrm R\\kern-.15em\\lower.5ex\\hbox{S}}");
    break;
  case 0x1f:
    fprintf(fptr[1],
            "{\\sevenrm U\\kern-.15em\\lower.5ex\\hbox{S}}");
    break;
  case ' ':
    ++ space_counter;
    cptr1 = inc_buf_ptr(ptr->buffer);
    if (((char)*cptr1 != ' ') || (*cptr1 == EOF))
      {
        if (ptr->flag == 0)
          fprintf(fptr[1], "{%s\\kern%3.3fem}", TextModeFont,
                  (float) SPACE * (float) space_counter);
        else
          fprintf(fptr[1], "\\kern%3.3fem ",
                  (float) SPACE * (float) space_counter);
        space_counter = 0;
```

```
    }
  break;
case '"':
  fprintf(fptr[1], "{\\tt \"}");
  break;
case '#':
  fprintf(fptr[1], "{\\tt\\#}");
  break;
case '$':
  fprintf(fptr[1], "{\\tt\\$}");
  break;
case '%':
  fprintf(fptr[1], "{\\tt\\%c}", (char) ptr->character);
  break;
case '&':
  fprintf(fptr[1], "{\\tt\\&}");
  break;
case '*':
  fprintf(fptr[1], "{\\tt *}");
  break;
case '-':
  fprintf(fptr[1], "{\\tt -}");
  break;
case '/':
  fprintf(fptr[1], "{\\tt /}");
  break;
case '<':
  fprintf(fptr[1], "{\\tt <}");
  break;
case '>':
  fprintf(fptr[1], "{\\tt >}");
  break;
case '\\':
  fprintf(fptr[1], "{\\tt\\char92}");
  break;
case '^':
  fprintf(fptr[1], "{\\tt\\char'136}");
  break;
case '_':
  fprintf(fptr[1], "{\\tt\\_\\kern.141em}");
  break;
case '{':
  fprintf(fptr[1], "{\\tt\\char'173}");
  break;
case '|':
  fprintf(fptr[1], "{\\tt |}");
  break;
case '}':
  fprintf(fptr[1], "{\\tt\\char'175}");
  break;
case '~':
  fprintf(fptr[1], "{\\tt\\char'176}");
  break;
```

```
            case 0x7f:
              fprintf(fptr[1],
                      "{\\sevenrm D\\kern-.15em\\lower.5ex\\hbox{T}}");
              break;
            default:
              /* text and bold face mode                                          */
              if ((ptr->flag == 0) && (bf_flag != 0))
                {
                  --bf_flag;
                  /* in case TextModeFont = \tt                                   */
                  if ((*TextModeFont == '\\')
                      && (*(TextModeFont + 1) == 't')
                      && (*(TextModeFont + 2) == 't'))
                    {
                      /* text mode bold face font is created by overprinting the slightly  */
                      /* shifted same character                                   */
                      fprintf(fptr[1], "%c", (char) ptr->character);
                      switch (ptr->character)
                        {
                        case 'e':
                          fprintf(fptr[1], "\\kern-.445em %c\\kern-.055em ",
                                  (char) ptr->character);
                          break;
                        case 'n':
                          fprintf(fptr[1], "\\kern-.46em %c\\kern-.04em ",
                                  (char) ptr->character);
                          break;
                        case 't':
                          fprintf(fptr[1], "\\kern-.445em %c\\kern-.055em ",
                                  (char) ptr->character);
                          break;
                        case 'u':
                          fprintf(fptr[1], "\\kern-.46em %c\\kern-.04em ",
                                  (char) ptr->character);
                          break;
                        default:
                          fprintf(fptr[1], "\\kern-.455em %c\\kern-.045em ",
                                  (char) ptr->character);
                        }
                    }
                  else
                    /* in case TextModeFont != \tt                                */
                    {
                      /* in case TextModeFont = \bf                               */
                      if ((*TextModeFont == '\\')
                          && (*(TextModeFont + 1) == 'b')
                          && (*(TextModeFont + 2) == 'f'))
                        {
#ifdef ASCII
                          /* ASCII JTEX                                           */
                          fprintf(fptr[1], "{\\rm\\mc %c}",
                                  (char) ptr->character);
#else
```

```
#ifdef NTT
                            /* NTT JTEX                                                  */
                            fprintf(fptr[1], "{\\rm\\dm %c}",
                                    (char) ptr->character);
#else
                            /* TEX                                                       */
                            fprintf(fptr[1], "{\\rm %c}",
                                    (char) ptr->character);
#endif
#endif
                          }
                        else
                          /* in case TextModeFont != \bf                                 */
                          {
#ifdef ASCII
                            /* ASCII JTEX                                                */
                            fprintf(fptr[1], "{\\bf\\gt %c}",
                                    (char) ptr->character);
#else
#ifdef NTT
                            /* NTT JTEX                                                  */
                            fprintf(fptr[1], "{\\bf\\dg %c}",
                                    (char) ptr->character);
#else
                            /* TEX                                                       */
                            fprintf(fptr[1], "{\\bf %c}",
                                    (char) ptr->character);
#endif
#endif
                          }
                      }
                  }
                /* non-text or non-bold face mode                                        */
                else
                  {
                    fprintf(fptr[1], "%c", (char) ptr->character);
                  }
              }
            ++char_counter;
            if ((char_counter >= 100) && (warn_flag2 <= 2))
              {
                ++warn_flag2;
#ifdef DEBUGGING
                printf("text2tex(): warn_flag2 is set to %d\n", warn_flag2);
#endif
                if (warn_flag2 > 2)
                  {
                    if (warn_flag1 <= 5)
                      fprintf(stderr, "\n");
                    fprintf(stderr,
                        "Warning: source file contains very long lines;\n");
                    fprintf(stderr,
                        "         their tails are sometimes truncated\n");
```

```
                }
            }
          if (((C_flag != 0) || (F77_flag != 0) || (PAS_flag != 0))
              && ((char) ptr->character == 0x27))
            {
              ++qt_counter;
              qt_counter %= 2;
            }
          if (((BAS_flag != 0) || (C_flag != 0))
              && ((char) ptr->character == '"'))
            {
              ++dqt_counter;
              dqt_counter %= 2;
            }
          continue;
        }
```

/* ———————————————— TEX mode ———————————————— */
/* Unfortunately, some languages cannot coexist with TEX. So, we sometimes have to modify input data and translate them into TEX. */

```
      if (ptr->flag == 2)
        {
          switch (ptr->character)
            {
              /* skip BASIC and COBOL line number                     */
            case '\n':
              fprintf(fptr[1], "\n");
              ++line_counter;
              ++page_len;
              char_counter = -1;
              cptr1 = inc_buf_ptr(ptr->buffer);
              cptr2 = inc_buf_ptr(cptr1);
              if (((BAS_flag != 0) || (CBL_flag != 0))
                   && ((*cptr1 <= ' ')
                       || ((*cptr1 >= '0') && (*cptr1 <= '9'))))
                {
#ifdef DEBUGGING
                  printf("skipping BASIC and COBOL line number ...\n");
                  printf("%c%c...\n", *cptr1, *cptr2);
#endif
                  while (*cptr1 != EOF)
                    {
                      cptr1 = fgetc2buffer(fptr);
                      cptr2 = inc_buf_ptr(cptr1);
                      if ((*cptr1 <= ' ') &&
                          (((*cptr2 > ' ') && (*cptr2 < '0'))
                           || (*cptr2 > '9')))
                        {
                          if (BAS_flag != 0)
                            *cptr1 = '\n';
                          break;
                        }
```

```
        }
    }
  break;
  /* skip MAPLE or MATLAB comment sign                                    */
case '#':
  if (((MAKE_flag == 0) && (PERL_flag == 0)
       && (SH_flag == 0) && (TCL_flag == 0)
       && (MAP_flag == 0) && (MLAB_flag == 0))
      || (char_counter != 0))
    fprintf(fptr[1], "%c", (char) ptr->character);
  break;
  /* skip REDUCE or OCTAVE comment sign                                   */
case '%':
  if (((RED_flag == 0) && (MLAB_flag == 0))
      || (char_counter != 0))
    fprintf(fptr[1], "%c", (char) ptr->character);
  break;
  /* skip BASIC comment sign                                              */
case 0x27:
  if ((BAS_flag == 0) || (char_counter != 0))
    fprintf(fptr[1], "%c", (char) ptr->character);
  break;
  /* skip FORTRAN comment sign                                            */
case '*':
  if (((F77_flag == 0) && (CBL_flag == 0))
      || (char_counter != 0))
    fprintf(fptr[1], "*");
  break;
  /* skip C++ or JAVA comment sign                                        */
case '/':
  if (((C_flag != 0) || (JAVA_flag != 0))
      && (char_counter == 0))
    {
      cptr1 = inc_buf_ptr(ptr->buffer);
      if (*cptr1 == '/')
        *cptr1 = ' ';
    }
  else
    fprintf(fptr[1], "/");
  break;
  /* skip LISP comment sign                                               */
case ';':
  if ((LISP_flag != 0) && (char_counter == 0))
    {
      cptr1 = inc_buf_ptr(ptr->buffer);
      while (*cptr1 == ';')
        {
          *cptr1 = ' ';
          cptr1 = inc_buf_ptr(cptr1);
        }
    }
  else
    fprintf(fptr[1], ";");
```

```
          break;
          /* skip FORTRAN comment sign                                 */
        case 'C':
          if ((F77_flag == 0) || (char_counter != 0))
            fprintf(fptr[1], "C");
          break;
          /* skip BASIC comment sign                                   */
        case 'R':
          cptr1 = inc_buf_ptr(ptr->buffer);
          cptr2 = inc_buf_ptr(cptr1);
          if ((BAS_flag != 0) && (char_counter == 0)
              &&((*cptr1 == 'E') || (*cptr1 == 'e'))
              &&((*cptr2 == 'M') || (*cptr2 == 'm')))
            {
              *cptr1 = ' ';
              *cptr2 = ' ';
            }
          else
            fprintf(fptr[1],"R");
          break;
          /* skip FORTRAN comment sign                                 */
        case 'c':
          if ((F77_flag == 0) || (char_counter != 0))
            fprintf(fptr[1], "c");
          break;
          /* skip BASIC comment sign                                   */
        case 'r':
          cptr1 = inc_buf_ptr(ptr->buffer);
          cptr2 = inc_buf_ptr(cptr1);
          if ((BAS_flag != 0) && (char_counter == 0)
              &&((*cptr1 == 'E') || (*cptr1 == 'e'))
              &&((*cptr2 == 'M') || (*cptr2 == 'm')))
            {
              *cptr1 = ' ';
              *cptr2 = ' ';
            }
          else
            fprintf(fptr[1],"r");
          break;
        default:
          fprintf(fptr[1], "%c", (char) ptr->character);
        }
      ++char_counter;
      continue;
    }
  }

/* ________________________ closing message ________________________ */
    fprintf(fptr[1], "\n\n");
#ifndef PLAIN
#ifdef LATEX            /*                          LaTeX             */
#ifdef ASCII            /* ASCII JTeX                                  */
    fprintf(fptr[1], "\\rm\\mc\n\n");
```

```
#else
#ifdef NTT              /* NTT JTeX                                     */
    fprintf(fptr[1], "\\rm\\dm\n\n");
#else
    fprintf(fptr[1], "\\rm\n\n");
#endif
#endif
    fprintf(fptr[1], "\\end{document}\n");
#else                   /*              plain TeX                       */
#ifdef ASCII            /* ASCII JTeX                                   */
    fprintf(fptr[1], "\\rm\\mc\n\n");
#else
#ifdef NTT              /* NTT JTeX                                     */
    fprintf(fptr[1], "\\rm\\dm\n\n");
#else
    fprintf(fptr[1], "\\rm\n\n");
#endif
#endif
    fprintf(fptr[1], "\\bye\n");
#endif
#endif
#ifndef DEBUGGING       /* last message                                 */
    fprintf(stderr, "... done\n");
#endif
}
```