```
/* ____________________________________________________________________ *
```

src2tex* version 2.12 (Sep 9, 1996)

KAZUO AMANO† AND SHINICHI NOMOTO‡

```
* ____________________________________________________________________ */

/* ____________________________________ getdata.c ____________________________________ */

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

/* ____________________________________ buffer storage ____________________________________ */
/* Input data are temporarily stored in buffer storage
        Buffer[0] Buffer[1] ··· Buffer[BUFFER_SIZE-1] .
For further details, see the definitions of functions of getdata.c, langflag.c, modflag.c and text2tex.c .    */

int Buffer[BUFFER_SIZE];

/* ____________________________________ increase buffer pointer buf_ptr ____________________________________ */
/* We access buffer storage by using integer pointer to Buffer[], which we call buffer pointer. This function
simply increases buffer pointer provided that Buffer[BUFFER_SIZE] is identified with Buffer[0].    */

int *inc_buf_ptr(buf_ptr)
int *buf_ptr;
{
  if ((buf_ptr == Buffer -1) || (buf_ptr == Buffer + BUFFER_SIZE))
    {
      fprintf(stderr, "\nError: buffering error !!\n");
      exit(EXIT_FAILURE);
    }
  if (++buf_ptr == Buffer + BUFFER_SIZE)
    buf_ptr = Buffer;
  return buf_ptr;
}

/* ____________________________________ decrease buffer pointer buf_ptr ____________________________________ */
```

* Permission to use, copy, and modify this software and its documentation is granted under no conditions. However, the authors would be very happy if users could inform any modifications to kamano@tansei.cc.u-tokyo.ac.jp. Since src2tex is a free software, there is no warranty of any kind for the program.

† 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)

/* This function is an inverse function of the above inc_buf_ptr(). In fact, dec_buf_ptr() decreases buffer pointer on the assumption that Buffer[-1] is identified with Buffer[BUFFER_SIZE-1]. */

```
int *dec_buf_ptr(buf_ptr)
int *buf_ptr;
{
  if ((buf_ptr == Buffer -1) || (buf_ptr == Buffer + BUFFER_SIZE))
    {
      fprintf(stderr, "\nError: buffering error !!\n");
      exit(EXIT_FAILURE);
    }
  if (buf_ptr-- == Buffer)
    buf_ptr += BUFFER_SIZE;
  return buf_ptr;
}
```

/* ———————————————— fgetc to buffer Buffer[] ———————————————— */
/* In case of first call, *i.e.*, in case of stat_flag=0, fgetc2buffer() increases stat_flag and fills $\frac{1}{2}$ of buffer

Buffer[0] Buffer[1] ··· Buffer[(int)(BUFFER_SIZE/2)-1]

with input data which we fgetc() from a file given on the command line. After that, fgetc2buffer() put

buf_ptr1=Buffer, buf_ptr2=Buffer+(int)(BUFFER_SIZE/2)

and returns buf_ptr1. When stat_flag$\neq$ 0, fgetc2buffer() reads a datum from a given file, substitutes it to *buf_ptr2 and increases buf_ptr2. And also, it increases buffer pointer buf_ptr1 and returns buf_ptr1. So, fgetc2buffer() uses a half of Buffer[] for input data and the remaining half for input history. */

```
int *fgetc2buffer(fptr)
FILE *fptr[];
{
    static int stat_flag = 0;
    static int *buf_ptr1, *buf_ptr2;
    int i;

    if (stat_flag == 0)
      {
          ++stat_flag;
          for (i = 0; i < (int) (BUFFER_SIZE / 2); ++i)
              Buffer[i] = fgetc(fptr[0]);
          for (i = (int) (BUFFER_SIZE / 2); i < BUFFER_SIZE; ++i)
              Buffer[i] = 0x00;
          buf_ptr1 = Buffer;
          buf_ptr2 = Buffer + (int) (BUFFER_SIZE / 2);
          return buf_ptr1;
      }
    else
      {
          *buf_ptr2 = fgetc(fptr[0]);
          buf_ptr2 = inc_buf_ptr(buf_ptr2);
          buf_ptr1 = inc_buf_ptr(buf_ptr1);
          return buf_ptr1;
      }
```

```
}
```