Fine soccer(6)

Morio Kikuchi

Abstract :

We modify the present rule of soccer in order to be able to get points of plural kinds referring to the point system of rugby.

1. Pitching machine

Pitching machine has not been as made use of as baseball in the traditional rugby. The plays in which pitching machine can be used will be about lineout, high punt and mark. Line out is unsuitable because of its high release point of ball and I do not think that there is a need of high cost on high punt and mark because the importance of them will be not so high.

We must devise initial velocity vector of a ball in order for a friend's side player to be able to catch the ball in symmetry kick which was introduced the last. When a ball is dropped near kicker, even if the velocity is small and there is a little deviation in meridian angle and azimuth, a friend's side player can catch the ball. However, when we try to make a far player catch the ball, high precision is required. If there is no an enemy's side player's mark to a far player, even if the velocity is insufficient a little, the player can catch the ball if kicker takes care of meridian angle and azimuth. However, if there is a mark, we must make the velocity larger and take meridian angle which corresponds to this velocity. I think that probably, there is a limit for a ball to be able to be catched in symmetry kick. Formation will be composed after recognition of the limit.

A ball is projected with kick right after release of ball from hands or drop kick in symmetry kick. If a ball goes near kicker, the control is easy, however, if a ball goes a long way from kicker, the control becomes difficult. In the case that we take a practice in which a game is resumed with symmetry kick, a ball must reaches a catcher each time. Therefore, on projection of ball, pitching machine will be more efficient than human's kick.

2. Teacup type pitching machine

We consider pitching machine which looks like motion mechanism of a teacup which is in amusement park, etc. Figure 1 shows mechanical conception figure of teacup type pitching machine. The first rotor is rotated with a fixed motor. In the first rotor, a dynamo and motor of which power source is the dynamo are installed. If the first rotor rotates, the dynamo produces power and because the motor connects to the second rotor by means of a drive shaft, if the first rotor rotates, the second rotor rotates too. Right after beginning of rotation, because the dynamo's power is low, there is a need of doing a device that rugby ball is fixed temporarily and stably.

Release of rugby ball is done with a dynamo and a solenoid which are installed in the second rotor. Figure 2 shows an example of release mechanism. The release is done when axis of rugby ball becomes almost at right angles to the first rotor's rotation circle. Actually, we design the syncronization adjusting rotation of the motor which is installed on the first rotor.

In Figure 2, the spin of projected rugby ball is in horizontal. If we want to take the spin to the direction of initial velocity, like Figure 3, rotating rollers which are provided with the second rotor by a drive shaft which is fixed on the first rotor, we reverse the second rotor at the same rate as the first rotor. In the left figure, the rollers and drive system which connects to the drive shaft are omitted. The right figure shows a cross section of rugby ball and the rollers. However, there is a possibility that axis

of the rollers does not becomes in the same direction as axis of the rugby ball. As we understand from the figure, the minimum value of stroke of the stopper namely solenoid is decided from the arrangement of the rollers.

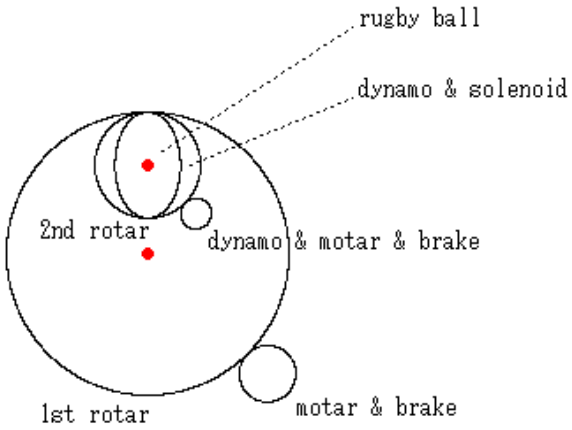

Figure 1

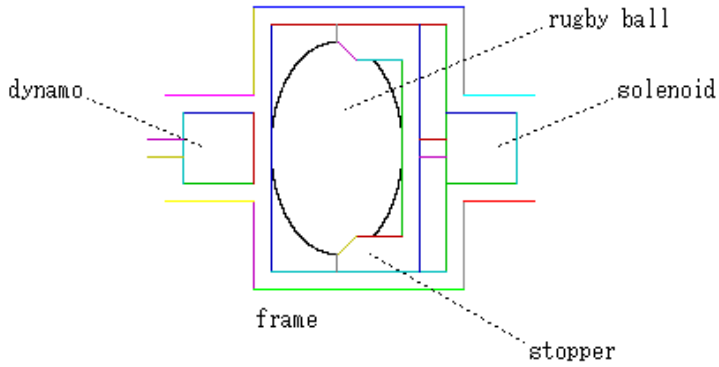

Figure 2

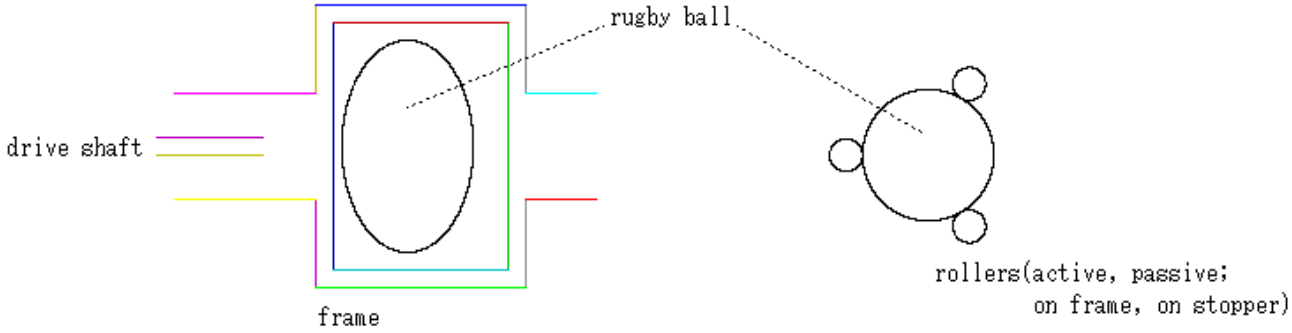

Figure 3

3. Uncontact type switch

Because the second rotor rotates, uncontact type switch is needed in order to release rugby ball. Sending supersonic waves or laser beams to sensor of the second rotor as a signal and processing them, rugby ball is released at a given position angle of the first rotor.

4. Problem

We call the mean of the time which is required to release rugby ball mean release time. It is assumed that the time deviation from a mean release time is Δt and the angle of rotation of the first rotor for Δt is $\Delta\theta$. Figure 4 shows reception of a signal, beginning of release, end of release and $\Delta\theta$. In

the figure, reception of a signal is done at the lowest point of the first rotor assuming that velocity of rugby ball is not so high. For examle, In the case that the rotation radius r of the centre of rugby ball is 0.5m, the velocity v of rugby ball is 20m/s and Δt is 0.001sec, $\Delta\theta$ becomes as follows:

$$\omega = \overline{dt}\theta, \quad \Delta\theta = \omega\Delta t$$
$$v = r\omega = 0.5 \cdot \omega = 20, \quad \omega = 40$$
$$\Delta\theta = 40 \cdot \Delta t[\text{rad}] = 40 \cdot \Delta t \cdot 360/(2\pi)[\text{deg}] \qquad 2300 \cdot \Delta t = 2.3$$

There is almost no deviation in azimuth of initial velocity of rugby ball, however, the coefficient of Δt is large like the above in deviation in meridian angle. Besides, not only pitching machine side but condition of rugby ball affects it. For example, If surface of rugby ball is wet, it will lead to $\Delta t < 0$.

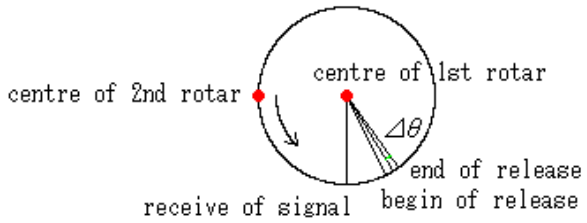

Figure 4

****

ファインサッカー (6)

菊池盛雄

アブストラクト：
ラグビーの得点体系を参考にして複数の得点が入るように現在のサッカーのルールを修正します。

1. ピッチングマシン

従来のラグビーでは野球ほどピッチングマシンが活用されていません。ピッチングマシンを用いることができるプレイは、ラインアウト、ハイパント、マークぐらいでしょう。ラインアウトはボールのリリースポイントが高すぎて不向きであり、ハイパント、マークはプレイの重要度がそれほど高くないでしょうから高いコストをかける必要があるとは思われません。

前回紹介したシンメトリーキックは味方のプレーヤーがボールをキャッチできるようにボールの初速度ベクトルを工夫しなければなりません。キッカーの近くにボールを落とすのであれば、たとえ速度が小さく、蹴射角、方向角に多少ぶれがあっても、味方のプレーヤーがボールをキャッチできます。しかし、far なプレーヤーにボールを取らせようとすると、高い精度が要求されます。もし、far なプレーヤーに敵方のプレーヤーのマークがなければ、速度が多少足りなくてもキッカーが蹴射角、方向角に気をつければボールをキャッチできます。しかし、マークがあれば速度を大きくし、かつこの速度に対応する蹴射角にしなければなりません。おそらく、シンメトリーキックにはボールをキャッチできる限界があると思われます。フォーメーションはこの限界を認識した上で構成されることになるでしょう。

シンメトリーキックでは手放しキックまたはドロップキックでボールを蹴射します。ボールがキッカーに近ければコントロールが容易ですが遠ければコントロールが困難になります。シンメトリーキックで試合を再開する練習をする場合は、毎回ボールが正確にキャッチャーに到達しなければなりません。ですから、ボールは人間が蹴射するのではなくピッチングマシンが投射する方が効率的でしょう。

2. ティーカップ型ピッチングマシン

遊園地などにあるティーカップの運動機構に似たピッチングマシンを考察します。図 1 はティーカップ型ピッチングマシンの機械的概念図です。第一ローターは固定モータで回します。第一ローターにはダイナモとこのダイナモを電力源とするモーターが据え付けられています。第一ローターが回転すればダイナモは電力を発生し、このモーターと第二ローターはドライブシャフトでつながっているので、第一ローターが回転すれば第二ローターも回転します。第一ローターが回転を開始した直後はダイナモの電力が小さいので、ラグビボールを暫時安定的に第二ローターに固定する工夫を施す必要があります。

ラグビボールの解放は第二ローターに据え付けられているダイナモとソレノイドで行います。図 2 は解放機構の例です。解放はラグビーボールの軸と第一ローターの回転円が直角に近くなった時に行います。実際には第一ローターに据え付けられているモーターの回転を調整してこの同期を図ります。

図 2 では蹴射されたラグビーボールのスピンは水平面内にあります。蹴射されたラグビーボールのスピンを初速度の方向に取りたい場合は、図 3 のように第一ローターに固定したドライブシャフトで第二ローターに設けたローラーを回し、第二ローターを第一ローターと同じ回転数で逆回転させます。左図ではローラーおよびドライブシャフトにつながる駆動系を省略しています。右図はラグビーボールとローラーの断面を示しています。ただし、ローラーの軸がラグビーボールの軸と同方向とならない可能性があります。図からわかるようにローラーの配置からストッパーすなわちソレノイドのストロークの最小値が決まります。

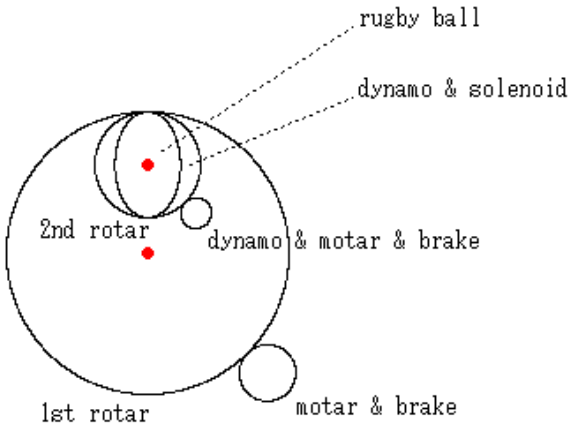

図 1

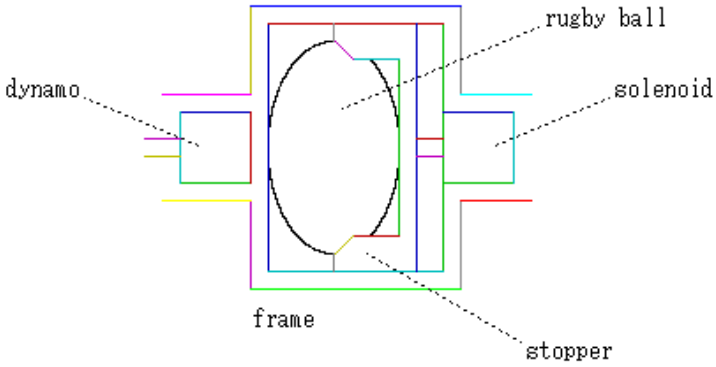

図 2

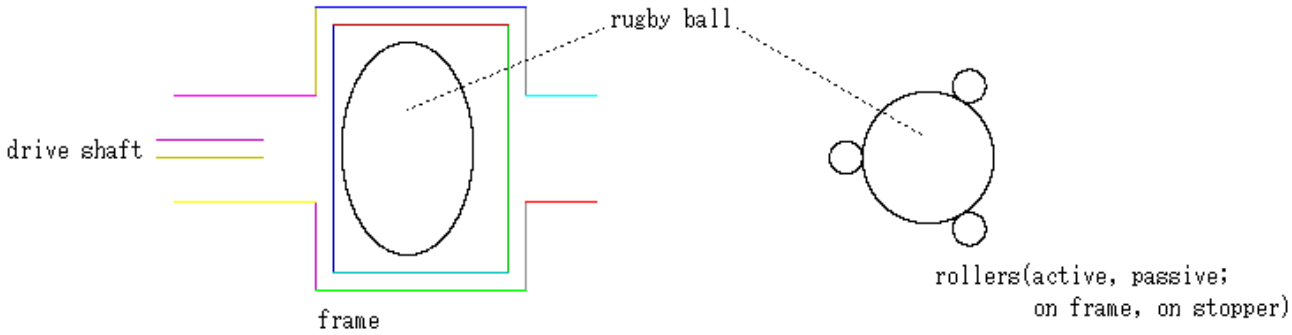

図 3

3. 非接触型スイッチ

第二ローターは回転しているのでラグビーボールを解放するためには非接触型スイッチを用いる必要があります。超音波、レーザー光をシグナルとして第二ローターのセンサーに送り、これを処理して第一ローターの所定の位置角でラグビーボールを解放します。

4. 課題

ラグビーボールの解放に要する時間の平均を平均解放時間と称します。平均解放時間からの時間的ぶれを Δt とし、Δt の間に第一ローターが回転する角を $\Delta\theta$ とします。図 4 はシグナルの受信、解放開始、解放終了、$\Delta\theta$ を示しています。図ではラグビーボールの速度があまり大きくないとしてシグナルの受信を第一ローターの最下点で行います。たとえば、ラグビーボールの中心の回転半径 r が 0.5m、ラグビーボールの速度 v が 20m/s、Δt が 0.001sec の場合の $\Delta\theta$ を求めます。

$$\omega = \overline{dt}\theta, \quad \Delta\theta = \omega\Delta t$$

$v = r\omega = 0.5 \cdot \omega = 20,\ \ \omega = 40$

$\Delta\theta = 40 \cdot \Delta t[\mathrm{rad}] = 40 \cdot \Delta t \cdot 360/(2\pi)[\mathrm{deg}] \quad 2300 \cdot \Delta t = 2.3$

ラグビーボールの初速度の方位角にはほとんどぶれがないのですが、子午線角のぶれにおいてはこのように Δt の係数が大きくなっています。また、ピッチングマシン側だけでなくラグビーボールの状態も影響を与えます。たとえば、ラグビーボールの表面がぬれていれば $\Delta t < 0$ となるでしょう。

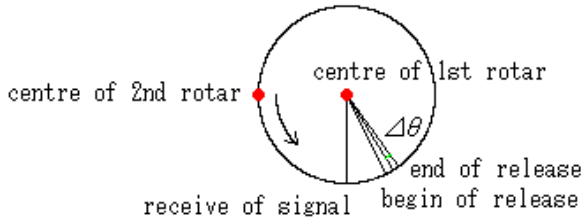

図 4

****

List 1:fgrep.c

```
/* fgrep program for EUC                                                    */
/* fgrep.c                                                                  */
/* by Morio Kikuchi 2018.1.1                                                */
/* WINDOW SYSTEM:X                                                          */
/* COMPILER:gcc 3.2.3                                                       */
/* COMMANDLINE:gcc -I/usr/X11R6/include fgrep.c                             */
/*                 -o fgrep -L/usr/X11R6/lib -lX11 -lm                      */
/* USAGE:fgrep String Place -w-(+) -i-(+) -d-(+) > ttt.bin                  */
/* Place:text files only(.exe, .obj, .bin are skipped)                      */
/* Place!=\(c:\)                                                            */
/* Line number is under columns:92, kinsoku:off                             */

#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include <X11/Xlocale.h>
#include <X11/cursorfont.h>
#include <X11/keysym.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <dirent.h>

#define GRP_or_EDT 1                /* 0, 1 */
#define FF_2 1                      /* 0, 1, 2, 3 */
#define T_TT /*0*/1

#define VGACOLORS 16
#define TRANS (65535./255)
#if GRP_or_EDT==0 || FF_2%2==0      /* 0, 2 */
#define LINEMODE 0
#else
#define LINEMODE 1
#endif
#define ROW_L_MIN 4
#define CD 38                       /* COLUMN_DIALOG */
#define COLUMN_MIN 40
#define DI 2
#define DI_ 1
#define DI_d (DI+2)                 /* d : dialog */
```

```
#define DJ_d (DJ+2)
#define DI_m 1
#define MAX_PATH 256
#define ASIZE (MAX_PATH+1)
#define ASIZEM (ASIZE-1)
#define dummy_R /*0x0d*//*0x0d*/'R'
#define dummy_T /*0x0d*//*0x1f*/'T'
#define dummy_E /*0x0d*//*0x0c*/'E'
#define NEST 0
#define NKF 0                       /* 1 : only X */
#define QMAX 1
#define XDSDY dv
#define FSIZE 14                    /* fontsize */
#define MONTH 0                     /* monthcheck in filer */
#define NOTEKBD /*0*/1              /* Shift+BS=Del */

#if T_TT==0
#define DJCSR 2                     /* cursordown in filer */
#define FCSRSIZE 50                 /* cursorsize in filer */
#define SPCNUM 6                    /* spacecheck in filer(t.c) */
#else
#define DJCSR 1                     /* cursordown in filer */
#define FCSRSIZE 46                 /* cursorsize in filer */
#define SPCNUM 7                    /* spacecheck in filer(tt.c) */
#endif

#define DSHIFT_2 2
#define SPC1 (0xa1)                 /* 1st byte of full space */
#define SPC2 (0xa1)                 /* 2nd byte of full space */

int XRESO,YRESO,WB,COLUMN,ROW_L,ROW_S,ROW,FMAX,DJ,UDX,UDY,dh,dv,
    CSRDY,ACTIVE,INACTIVE,RTC,RETURN,TABCOLOR,CC,CSRCOLOR,CSRCOLOR_FILER,
    TAB_c,AINDENT,DX_FRAME,DY_FRAME,DY_CAPTION,DY_MENU,DY_TOOLBAR,
    RIGHT_M,LEFT_m,AVMEMDENO;

char cut,paste,dialogflag,refflag,use_selector_flag,tabspaces,
     uflag,delorbs,unlinkflag,charflag,function,menuflag,d_or_t,usflag,nestflag,
     okflag_1,okflag_BL,okflag_w,bitbltflag,ROWflag,filerskip,
     BitBltflag,reffunc_global,allflag,lumpflag,lumpflag_dialog,puts_mline_flag,
     filerflag,nocloseflag,passflag,divideflag,divideflag_,dialogflag_REF,
     divisionnumber,mlinecolor,noclearflag,cqflag,deletedflag,BitBltflag_,
     refill_old,nobeepflag,flag_global,flag_2nd,u_s_flag,systemflag,indicationflag,
     nest_free_flag,reffunc_REF,reffunc_REP,l_s_flag,lumpflag_global,rependflag,
     restoreflag,filer_execute,filer_execute_phantom,newopen,redoskip,driveflag,
     insorover,Flag_k,beginjumpflag,linelength_new,overwriteflag,
     noelineflag,no_extraline,wsearch,to_sub;
```

```
unsigned char charcode,direction,direction_old,yorn;
int refill,icsr,jcsr,icsr_from,jcsr_from,icsr_last,icsr_global,delsp,icsr_f,jcsr_f,
    fn,fn_1st,fn_2nd,fsp,ftp,sizeoffname,jcsr_select,nest,cdflag,messageflag,argc_,
    jcsr_floor,icsr_filer,jcsr_filer,icsr_ref,jcsr_ref,icsr_filename,jcsr_filename,
    icsr_jump,jcsr_jump,icsr_program,jcsr_program,jcsrmax,sizeoffname_max=(CD+1)-8,
    spaces,start_of_arg,platform,icsr_cfg,jcsr_cfg,icsr_string,jcsr_string,
    csrcolor,asct,dsct,MOVEcsr,arraycheckflag;
long kceiltmp,kmax_dialog,k_from,k_to,dk,dk_old,dk_line,k_from_rep,k_to_rep,
     dk_word,dk_file,dk_ins,dk_cut,repcount,cfgmax,count_dir,count_file,topp_floor,
     firstk_filer,firstk_ref,firstk_filename,firstk_cfg,firstk_string,firstk_jump,
     firstk_program,firstk,firstk_from,line_end,member_last,member_global,
     firstk_dialog,kmax_sdb,mfsize,fsize_g,kmax_ml,k_g;
double both=2;

char **argv_,*openmode,*editflag,cc[]="@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\\]^_",
     fname_bg[ASIZE];
char appliname[]="BLE",fs1[ASIZE];
char *fs2[5]={
              "-*-fixed-medium-r-normal--14-*",
              "-*-mincho-medium-r-normal--14-*",
              "-*-gothic-medium-r-normal--14-*",
              "-*-*-medium-r-normal--14-*",
              "none"
              };  /* fontsize */
unsigned char **p,**fnames,*ptmp,*ptmp_line,*buf_line,*pcfg,linestring[11],
              ptmp_word[ASIZE],fname[ASIZE],mline[ASIZE+60],stock_db[ASIZE],
              ins[ASIZE],file_SA[ASIZE],stack_del[ASIZE],stack_bs[ASIZE],
              array[ASIZE],ref_s[ASIZE],rep_s[ASIZE],rep_s_[ASIZE],
              ref_t[ASIZE],rep_t_[ASIZE],p_dialog[ASIZE],p_restore[ASIZE],
              home_global[ASIZE],home_global_GCD[ASIZE],/*home_euc[ASIZE],*/
              home_deleted[ASIZE],home_tmp[ASIZE],two[5][2],home_ref[ASIZE],
              nsc[3][ASIZE],GRP_line[200];
long *topp,*kmax,*kceil;
FILE *fp,*fpi,*fpf;

char dbflag,imm_restart_flag;
long dbsize;

XColor irgb[VGACOLORS];
typedef struct {
int back,fore;} bf;
bf bfset[]={{15,0},{0,15}};
typedef struct {
char flag;int number;} fs;
fs *fstack;
typedef struct {
```

```
int fn,icsr,jcsr;long firstk;} ft;
ft *ftable,ft_tmp;
static struct dirent *dps[100];
typedef struct {
unsigned char dir[ASIZE],file[ASIZE];int rtn;} df;

Display *d;
int screen,depth;
Colormap cmap;
Window rw,w;
XSizeHints sh;
GC gcdisplay;
Pixmap pmap1,pmap3;
XFontSet font_fs;
XFontStruct **info;
Cursor cursor;
XEvent event,event_;
XKeyEvent keyevent;
KeySym keysym,sym;

unsigned long mask;
char **flist,**mlist,*def;
int mcount,fontnum;
XIM ime;
XIMStyle style;
XIC ic;
Status st;

void mainroop(void),setcsrcolor(int),csr(void),autoindent(void),end_ble(void),
     csr_tab(char),centering_theline(void),centering_csr(void),keydowns_f2(void),
     use_selector(char),ref(void),filename(void),jump(void),program(void),
     divide_display(void),switch_division(char),restore_display(void),
     show_file(void),open_file(char),show_top(char),restore_3(char),
     close_file(void),close_all(void),write_3vals(int),read_3vals(int),
     insertion_dk(char,long),nest_free(void),save_all(char),
     deletion_dk(void),deletion_dk_lump(void),BL(void),YKP(char),
     to_stack(unsigned char),to_stack_2b(unsigned char,unsigned char),swap_BL(char),
     insertion_u(void),insertion_cc(unsigned char),FILE_ref_tmp(char),
     bitblt(char,int,int,int,int,int,int),BitBlt_full(void),BitBlt_nomline(void),
     use_subroop(void),use_filer(void),filer(void),use_deleted(char),deleted(void),
     stc(char,int,int,unsigned char *,int),setstccolor(int),
     stc(char,int,int,unsigned char *,int),extraline(char),
     message(int,char),puts_message(int),puts_(int,int,unsigned char *),
     putstr(char,int,int,char *),putstrings(void),fopen_successed(void),
     while_puts_show_str(char,int,int,int,unsigned char *),fload_failed(void),
     monitorline(char),while_puts_show_monitorline(char,int,int),setup(void),
```

```
     paint(char,int,int,int,int,int),cleardevice_(char,int,int,int,int),
     text_home(void),text_end(void),page_down(void),page_up(void),scan_BL(void),
     csr_column_home(void),csr_column_end(void),csr_row_home(void),indicator(char),
     csr_down(void),csr_up(void),csr_right(void),tailcheck(void),csr_row_end(void),
     initpalette(void),closegraph_(void),kbhit_(void),delay_(long),beep(long),
     while_puts_theline(int),while_puts_fload_(char,int),arrange_colors(void),
     page_firstk(long),page_firstk_from(void),while_puts_show_(char,long),
     execute(unsigned char *),puts_mline(char,char *),half_line(char),
     BitBlt_indicator(void),restore_page_and_oldcsr(void),
     csr_to_1(void),csr_to_1_BL(char),edit_cfg(void),edit_tmpcfg(void),
     string_visible(void),file_attri(void),breaks(char),find_0x1a(char),
     initIME(void),overwrite(void),refind(char),hcentering(char),copy_cfg(void),
     prompt_cq(char),clear_topp(void),FILE_jump(char),frees(void),half_word(char),
     restore_another(void),find_word(char),restore_in_PAINT(void),
     copy_string(void),within_linemax(void),FILE_filename(void),
     move_and_paste(void),filer_t(void),filer_tt(void),

     title(char *),before_mainroop(char *),before_mainroop_(char *),
     after_mainroop(void),backspace_dialog(void),BitBlt_dialog(void),
     page_firstk_dialog(long),while_puts_show_dialog(long),clear_dialog(char),
     text_end_dialog(void),trim_dialog(void),restore_dialog(void),
     within_linemax_dialog(void),insertion_cc_dialog(unsigned char),
     csr_row_home_dialog(void),csr_row_end_dialog(void),csr_tab_dialog(char),
     tailcheck_dialog(void),page_down_dialog(void),page_up_dialog(void),
     left_keydowns_dialog(void),csr_right_dialog(void),
     page_firstk_dialog_dummy(long),

     before_mainroop_menu(char *),before_mainroop_menu_REP(char *),
     after_mainroop_menu(void),while_puts_show_menu(int,char,unsigned char *),
     BitBlt_menu(void),csr_column_home_menu(void),csr_column_end_menu(void),
     left_keydowns_menu(void);
char gettype_p(long),gettype_u(long),gettype_mline(long),gettype_ac(long),
     gettype(char,unsigned char,long,long),gettype_jp(long),fopen_(void),
     p_realloc(void),ptmp_realloc(void),pdata_increase(long,unsigned char *,long),
     insertion_dk_lump(long,long),left_keydowns(void),

     gettype_dialog(long),gettype_sdb(long),csr_left_dialog(void),

     gettype_fnames(int,long),gettype_buf(long,unsigned char *);
unsigned char subroop(void),*fnames_shortened(int);
int reference(char),replacement(long,long,long),reference_lump(char),
    replacement_lump(long,long,long),large_small(long,long),
    shorten(void),shorten_(void),fload(char),fsave(char,char),
    deletion(void),backspace(void),deletion_onlymem(void),
    text_to_file(char,char),file_to_text(void),GKS(KeySym),GKS_(long),
    scroll_down(char),scroll_up(char),csr_left(void),return_is(long),
```

```
    initgraph_(void),insertion(unsigned char),memory(char),YKP_word(char),
    wordcheck(char,long),wordcheck_kana(char,long),arraycheck(void),
    wordcheck_unvisible(char,long),spacecheck(long,long),
    while_puts_thepart(long,long),linestringcheck(void),read_cfg(int),
    getTAB(int),CopyFile(unsigned char *,unsigned char *),
    ff_fc(double),wordcheck_2bytes(char,char,long),
    ishead(long),ishead_(long,long),ishead_ac(long),notspacecheck(char,char *),

    deletion_dialog(void),scroll_down_dialog(void),scroll_up_dialog(void),
    insertion_dialog(unsigned char),ishead_dialog(long);
long top_icsr(int,int),getspan_u(void),
     get_firstk(long,long),get_kstart(long),max(long,long),min(long,long),
     while_puts_dline(long,long),while_puts_firstk(long,long),
     while_puts_linenumber(long,long),

     gethead_diaclog(char,long);

void imm_pause(void),imm_restart(void),imm_restart_filer(void),
     WM_func_CHAR(unsigned char),WM_funcIME_CHAR(unsigned char *),
     InputPosition(XIC,int,int);
XIMStyle InputStyle(XIM);
XIC InputContext(XIM,XIMStyle,XFontSet,Window);

int wndproc_filer(void),wndproc_ref(void),wndproc_deleted(void),wndproc_BL(void),
    wndproc(void);

void mnuproc_MULTIFILE(char *);
void mnuproc_REP(char *);
void dlgproc_OPEN(char);
void dlgproc_JUMP(void);
void dlgproc_SAVE(char);
void dlgproc_SAVE_(void);
void dlgproc_REN(void);
void dlgproc_INS(void);
void dlgproc_FILE(char);
void dlgproc_REF(char);
void dlgproc_REP(char);

df divide_plc(unsigned char *);


#if GRP_or_EDT==0
int main(int argc,unsigned char **argv)
{
int i,flag,len;
unsigned char buf[ASIZE],plc[ASIZE];
```

```
df dfset;

/*unlink("/root/gcc/cpage.bin");*/
getcwd(fname_bg,ASIZE);

initgraph_();
refill=1;
/*putenv("TZ=EST0EDT");
tzset();*/

function=1;
wsearch=0;
l_s_flag=0;
to_sub=0;

if(argc==1)      {printf(" No String\n");goto end;}
else if(argc==2) {printf(" No Place\n");goto end;}
else if(argc==3){
strcpy(nsc[0],argv[1]);
strcpy(nsc[1],argv[2]);
}
else{
strcpy(nsc[0],argv[1]);
strcpy(nsc[1],argv[2]);

for(i=3;i<argc;i++)
notspacecheck(0,argv[i]);
}

/*printf(" %s %s\n",argv[1],argv[2]);
printf(" %s %s\n",nsc[0],nsc[1]);*/

strcpy(ref_s,nsc[0]);
strcpy(plc,nsc[1]);
dfset=divide_plc(plc);

if(dfset.rtn==0){
strcpy(buf,dfset.dir);

if(strlen(dfset.file)==0){
len=strlen(buf);
if(buf[len-1]!='/') strcat(buf,"/.*");
else strcat(buf,".*");
}/**if(strlen(dfset.file)**/
else{
strcat(buf,dfset.file);
```

```
}/**else(strlen(dfset.file)**/

printf(" 0:String:%s Place:%s w:%d i:%d d:%d\n",nsc[0],buf,wsearch,l_s_flag,to_sub);
printf(" maximum filesize(mfsize):%ld[MB]\n\n",mfsize);
chdir(dfset.dir);
make_list(dfset.dir,dfset.file);
chdir(fname_bg);
}
else if(dfset.rtn==1){                /* e and r */
printf(" 1:String:%s Place:%s w:%d i:%d d:%d\n",nsc[0],plc,wsearch,l_s_flag,to_sub);
printf(" maximum filesize(mfsize):%ld[MB]\n\n",mfsize);
flag=fgrep(plc);
if(flag==0){
free(p[fn]);
fload_failed();
}
else if(flag==1){
fload_failed();
}
}
else printf(" 2:Bad Place\n");

end:
closegraph_();
return 0;
}/** main **/
#else
int main(int argc,unsigned char **argv)
{
unsigned char oldstring[ASIZE];

argc_=argc;
argv_=(char **)argv;

if(FF_2/2) getcwd(fname_bg,ASIZE);

initgraph_();
refill=1;

strcpy(fname,"");
strcpy(oldstring,"");

if(argc==1){
/* unit -> */
start:
```

```
beginjumpflag=0;
extraline(1);

dlgproc_OPEN(0);
if(refill==0) goto end;
if(fopen_()==1){
strcpy(fname,oldstring);puts_mline(0,"Reinput a filename.");goto start;}

fopen_successed();

icsr=0;jcsr=0;firstk=0;
if(fload(0)==1){
fload_failed();strcpy(fname,oldstring);
puts_mline(0,"Reinput a filename.");goto start;}
strcpy(fnames[fn],fname); editflag[fn]=0;
sizeoffname=max(strlen(fname),sizeoffname);
if(unlinkflag) unlink(fname);
/* <- unit */
}/**if(strlen(args))**/
else{
/*999*/
if(1) kbhit_();

strcpy(array,argv[1]);

if((arraycheckflag=arraycheck())>1) use_filer();

if(arraycheck()>0) {puts_mline(0,"Reinput a filename.");goto start;}
else strcpy(fname,array);

if(argc>2 && strlen(argv[2])<11){
strcpy(linestring,argv[2]);
if(linestringcheck()==0) beginjumpflag=1;
else firstk=0;
}
else firstk=0;

if(fopen_()==1) {puts_mline(0,"Reinput a filename.");goto start;}

fopen_successed();

icsr=0;jcsr=0;
if(fload(0)==1){
fload_failed();goto start;}
strcpy(fnames[fn],fname); editflag[fn]=0;
sizeoffname=max(strlen(fname),sizeoffname);
```

```
if(unlinkflag) unlink(fname);
/* <- unit */
}/**else(strlen(args))**/

write_3vals(ftp-1);
csr();                              /* after page_firstk() */
/*extraline(1);*/

mainroop();

end:
closegraph_();
return 0;
}/** main **/
#endif


int notspacecheck(char flag,char *p)
{
int i,j,len,notspacecount;
static int flag_=0,k=0;

len=strlen(p);

if(flag==0){
if(flag_==0){
for(i=0;i<len;i++){
if(GRP_or_EDT==0 && p[i]=='-'){

if(i+1<len && p[i+1]=='w'){
     if(i+2<len && (p[i+2]=='-' || p[i+2]=='0')) {wsearch=0;i+=3;}
else if(i+2<len && (p[i+2]=='+' || p[i+2]=='1')) {wsearch=1;i+=3;}
else if(i+2<len && (p[i+2]==' ' || p[i+2]=='1')) {wsearch=1;i+=2;}
else if(i+2==len)               {wsearch=1;i+=2;}
}
else if(i+1<len && p[i+1]=='i'){
     if(i+2<len && (p[i+2]=='-' || p[i+2]=='0')) {l_s_flag=0;i+=3;}
else if(i+2<len && (p[i+2]=='+' || p[i+2]=='1')) {l_s_flag=1;i+=3;}
else if(i+2<len && (p[i+2]==' ' || p[i+2]=='1')) {l_s_flag=1;i+=2;}
else if(i+2==len)               {l_s_flag=1;i+=2;}
}
else if(i+1<len && p[i+1]=='d'){
     if(i+2<len && (p[i+2]=='-' || p[i+2]=='0')) {to_sub=0;i+=3;}
else if(i+2<len && (p[i+2]=='+' || p[i+2]=='1')) {to_sub=1;i+=3;}
else if(i+2<len && (p[i+2]==' ' || p[i+2]=='1')) {to_sub=1;i+=2;}
else if(i+2==len)               {to_sub=1;i+=2;}
```

```
}

}/**if(p[i],'-')**/
else if(GRP_or_EDT==1 && p[i]=='\"'){
j=i+1;
while(1){
if(j==len) break;
else if(p[j]=='\"'){
if(j-(i+1)>0) {strncpy(nsc[k],&p[i+1],j-(i+1));nsc[k][j-(i+1)]='\0';k++;}
break;
}

j++;
}/**while(1)**/

i=j;
}/**if(p[i],'\"')**/
else if(GRP_or_EDT==1 && p[i]!=' '){
j=i+1;
while(1){
if(j==len || p[j]==' '){
strncpy(nsc[k],&p[i],j-i);nsc[k][j-i]='\0';k++;
break;
}

j++;
}/**while(1)**/

i=j;
}/**else if(p[i],!' ')**/
else{                                /* ' ' */
}
}/**for(i)**/

#if GRP_or_EDT==1
flag_++;
#endif
}/**if(flag_)**/

notspacecount=k;
}/**if(flag,flag_)**/
else if(flag==1){
notspacecount=strlen(nsc[0]);
}
else if(flag==2){
notspacecount=strlen(nsc[1]);
```

```
}

return notspacecount;
}/** notspacecheck **/


df divide_plc(unsigned char *plc)
{
char type;
int i,j,len;
unsigned char buf[ASIZE];
df dfset;

len=strlen(plc);

i=0;
while(1){
type=gettype_buf(i,plc);
if(type<=2){
if(plc[i]=='\\') plc[i]='/';

i++;
}
else{
i+=2;
}

if(i==len/*gth*/) break;
}

i=0;
while(1){
if(plc[i]=='*' || plc[i]=='?'){
j=i-1;
while(2){
if(j==-1) break;

if(plc[j]=='/'){
/*if(j==0 || isleadbyte(plc[j-1])==0) */break;
}

j--;
}/**while(2)**/
break;
}/**if(plc[i])**/
```

```
i++;if(i==len) break;
}/**while(1)**/

getcwd(buf,ASIZE);

/* 4 cases */
if(i==len){
if(chdir(plc)==0){
/* 1st:0, tmp;\ */
chdir(buf);
strcpy(dfset.dir,plc);  /* tmp;\ */
strcpy(dfset.file,"");
}
else{
/* 2nd:1, my.bak */
if(access(plc,0)==0 /*&& access(plc,4)==0*/) {dfset.rtn=1;goto end;}  /* e, r */
else {dfset.rtn=2;goto end;}
}
}/**if(i)**/
else{
if(j==-1){
/* 3rd:0, *.* */
if(buf[/*3*/1]!='\0') strcat(buf,"/");
strcpy(dfset.dir,buf);  /* c:\vc\ */
strcpy(dfset.file,plc);  /* *.* */
}
else{
/* 4th:0, \vc\*.* */
strcpy(dfset.file,&plc[j+1]);  /* *.* */
plc[j+1]='\0';
if(chdir(plc)==-1) {dfset.rtn=2;goto end;}
chdir(buf);
strcpy(dfset.dir,plc);  /* \vc\ */
/*printf(" %s %s\n",dfset.dir,dfset.file);*/
}
}/**else(i)**/

strcpy(array,dfset.dir);
arraycheck();
strcpy(dfset.dir,array);

dfset.rtn=0;

end:
return dfset;
}/** divide_plc **/
```

```
int strci(const char *str1,const char *str2)
{
int i;
long k1,k2;

k1=strlen(str1);
k2=strlen(str2);

if(k1>0 && k1==k2) {}
else{
if(k1>=k2) return 1;
else return -1;
}

i=0;
while(1){
if(i==/*n*/k1) break;

if(str1[i]>=/*0x41*/'A' && str1[i]<=/*0x5a*/'Z'){
if(str1[i]==str2[i] || str1[i]==str2[i]-0x20) i++;
else break;
}
else if(str1[i]>=/*0x61*/'a' && str1[i]<=/*0x7a*/'z'){
if(str1[i]==str2[i] || str1[i]==str2[i]+0x20) i++;
else break;
}
else{
if(str1[i]==str2[i]) i++;
else break;
}
}

if(i==/*n*/k1) return 0;
else{
if(k1>=k2) return 1;
else return -1;
}
}/** strci **/


int fgrep(unsigned char *str)
{
int sz;
unsigned char buf[5];
```

```
sz=strlen(str);
if(sz>4){
strcpy(buf,&str[sz-4]);
if(strci(".EXE",buf)==0) return 2;
if(strci(".OBJ",buf)==0) return 2;
if(strci(".BIN",buf)==0) return 2;
}

strcpy(array,str);
if(arraycheck()>0) return 2;
strcpy(fname,array);

if(fopen_()==1) return 2;

fopen_successed();

icsr=0;jcsr=0;firstk=0;
if(fload(0)==1){
fload_failed();return 1;}

/*printf(" %d %ld\n",fn,kmax[fn]);*/
reference((char)((wsearch>0)?0:1));

return 0;
}/** fgrep **/


void fprintf_(char* str1)
{
FILE *fp;

fp=fopen("/root/gcc/cpage.bin","ab");

fprintf(fp," %s\n",str1);

fclose(fp);
}/** fprintf_ **/


void fprintf_2(char* str1)
{
unsigned char home[ASIZE];
FILE *fp;

strcpy(home,home_global);
```

```
strcat(home,"cpage_f.bin");

fp=fopen(home,"wb");

fprintf(fp,"%s",str1);

fclose(fp);
}/** fprintf_2 **/


void Quick_Find(char flag_1,char flag_2)
{
if(flag_1==0) refind(0);
else if(flag_1==1){
  if(k_to-k_from<=ASIZEM-1){
  strncpy(ref_s,&ptmp_word[0],dk_word);ref_s[dk_word]='\0';strcpy(ref_t,ref_s);
  function=0;refind(flag_2);}
}
else if(flag_1==2){
  if(k_to-k_from>0 && k_to-k_from<=ASIZEM-1){
  strncpy(ref_s,&ptmp[0],dk);ref_s[dk]='\0';strcpy(ref_t,ref_s);
  function=0;refind(flag_2);}
}
}/** Quick_Find **/


void Replace(void)
{
char function_old;

  function_old=function;
  nest++;
  function=2;
  if(GKS_(ShiftMask)<0) dlgproc_REP(1);else dlgproc_REP(0);
  function=function_old+3;
  if(function==4) {charflag=1;nest--;}
  else nest=0;
}/** Replace **/


void Find(char flag)
{
  if(nest<NEST){
  }
  else nest_free();
}/** Find **/
```

```
int Enter(char flag)
{
if(flag==0){
  if(GKS_(ShiftMask)<0) {uflag=1;csr_row_home();}
  if(AINDENT==1) {if(GKS_(ControlMask)>=0) lumpflag=1;}
  else {if(GKS_(ControlMask)<0) lumpflag=1;}

  if(insertion('\n')==1) {lumpflag=0;uflag=0;goto end;}

  if(AINDENT==1) {if(GKS_(ControlMask)>=0) autoindent();}
  else {if(GKS_(ControlMask)<0) autoindent();}
  if(GKS_(ShiftMask)<0) uflag=0;

goto end_;
}
else{
  if(GKS_(ShiftMask)<0) {uflag=1;csr_row_home();}

  if(insertion('\n')==1) {uflag=0;goto end;}

  if(GKS_(ShiftMask)<0) uflag=0;

goto end_;
}

end:
return 1;

end_:
return 0;
}/** Enter **/


void putstrings(void)
{
int len;
char str[ASIZE];

len=sprintf(str,"ftp=%d fsp=%d fn=%d FMAX=%d",ftp,fsp,fn,FMAX);
if(divisionnumber<=1)
putstr(1,COLUMN-len+(DI_+RIGHT_M-1),ROW_L+2,str);
else if(divisionnumber==2)
putstr(1,COLUMN-len+(DI_+RIGHT_M-1),ROW_S+2,str);
}/** putstrings **/
```

```
void putstr(char flag,int x,int y,char *str)
{
while_puts_show_str(flag,0,x,y,str);
}/** putstr */


void initIME(void)
{
int width,height,h;
unsigned char buf[11];

setlocale(LC_ALL,"");
if(XSupportsLocale()==False) exit(1);
if(XSetLocaleModifiers("")==NULL) exit(1);
if((ime=XOpenIM(d,NULL,NULL,NULL))==NULL) exit(1);

style=InputStyle(ime);



/*strcpy(fs1,fs2[fontnum]);*/
if(0) strcpy(fs1,fs2[3]);
else{
/* scalable_ */
h=UDY+dh;
h=max(min(h,64),8);
/*printf(" UDX=%d\n",UDX);
printf(" UDY=%d\n",UDY);
printf(" dh=%d\n",dh);
printf(" h=%d\n",h);*/

if(fontnum==0)
strcpy(fs1,"-*-fixed-medium-r-normal--");
else if(fontnum==1)
strcpy(fs1,"-*-mincho-medium-r-normal--");
else if(fontnum==2)
strcpy(fs1,"-*-gothic-medium-r-normal--");
else
strcpy(fs1,"-*-*-medium-r-normal--");

/*itoa(abs(h),buf,10)*/gcvt(abs(h),3,buf);
strcat(fs1,buf);
strcat(fs1,"-*");
strcat(fs1,"-*-*-*-*-*-*");
```

```
}

font_fs=XCreateFontSet(d,fs1,&mlist,&mcount,&def);



ic=InputContext(ime,style,font_fs,w);
XGetICValues(ic,XNFilterEvents,&mask,NULL);
}/** initIME **/


void InputPosition(XIC ic,int icsr,int jcsr)
{
int dx,dy;
XPoint point;
XVaNestedList list;

if(style & XIMPreeditPosition){}
else return;

if(dialogflag>0) {dx=(icsr+DI_d)*UDX;dy=(jcsr+DJ_d)*UDY;}
else             {dx=(icsr+DI)*UDX;dy=(jcsr+DJ)*UDY;}
point.x=dx;
point.y=dy+FSIZE+DSHIFT_2;

list=XVaCreateNestedList(0,XNSpotLocation,&point,NULL);
XSetICValues(ic,XNPreeditAttributes,list,NULL);
XFree(list);
}/** InputPosition **/


XIMStyle InputStyle(XIM ime)
{
int i,j,k;
XIMStyles *ime_styles;
static XIMStyle preedit[]={XIMPreeditPosition,XIMPreeditArea,XIMPreeditNothing,0};
static XIMStyle status[]={XIMStatusArea,XIMStatusNothing,0};

XGetIMValues(ime,XNQueryInputStyle,&ime_styles,NULL);

i=0;
while(1){

j=0;
while(1){
```

```
for(k=0;k<ime_styles->count_styles;k++){
if((preedit[i] & ime_styles->supported_styles[k]) &&
   (status[j] & ime_styles->supported_styles[k]))
return ime_styles->supported_styles[k];
}

j++;
if(status[j]==0) break;
}

i++;
if(preedit[i]==0) break;
}

return 0;
}/** InputStyle **/


XIC InputContext(XIM ime,XIMStyle style,XFontSet font_fs_auto,Window w)
{
int dx,dy;
XIC ic;
XVaNestedList list;
XPoint point;

dx=(0+DI)*UDX;dy=(0+DJ)*UDY;
point.x=dx;
point.y=dy+FSIZE;

list=XVaCreateNestedList(0,XNFontSet,font_fs_auto,
                           XNSpotLocation,&point,NULL);
ic=XCreateIC(ime,XNInputStyle,style,
                 XNClientWindow,w,
                 XNPreeditAttributes,list,XNStatusAttributes,list,NULL);
XFree(list);
if(ic==NULL) exit(1);

return ic;
}/** InputContext **/


void hcentering(char flag)
{
char type;
char flag_,reallocflag;
int length,trim;
```

```
long member;
long k,k_right,k_left,k_0,dk,dk_centering;
long k1,k2,dk_deletion;

reallocflag=0;
flag_=0;

csr_row_end();
csr_tab(0);
if(icsr==0) return;
k=top_icsr(/*firstline+*/jcsr,icsr);

if(p[fn][k]=='\n' || k==kmax[fn]){
if(k==kmax[fn]){
lumpflag=1;
uflag=1;
if(insertion('\n')==1) {lumpflag=0;uflag=0;return;}
uflag=0;
lumpflag=0;
/*csr_up();csr_row_end();*/
}

csr_left();

while(1){
member=top_icsr(/*firstline+*/jcsr,icsr);
type=gettype_p(member);
if(type<=2) {if(p[fn][member]!=' ' && p[fn][member]!=0x09) break;}
else if(type==3){
if(p[fn][member]!=/*0xa1*/SPC1 || p[fn][member+1]!=/*0xa1*/SPC2) break;
}
else{}

if(icsr==0) return;

csr_left();
}

k_right=member;if(gettype_p(k_right)==3) k_right++;

csr_row_home();
while(1){
member=top_icsr(/*firstline+*/jcsr,icsr);
type=gettype_p(member);
if(type<=2) {if(p[fn][member]!=' ' && p[fn][member]!=0x09) break;}
else if(type==3){
```

```
if(p[fn][member]!=/*0xa1*/SPC1 || p[fn][member+1]!=/*0xa1*/SPC2) break;
}
else{}

csr_right();
}

k_left=member;

length=while_puts_thepart(k_left,k_right);
memmove(&buf_line[0],&p[fn][k_left],k_right-k_left+1);

k_0=top_icsr(/*firstline+*/jcsr,0);
trim=length-(k_right-k_left+1);
dk=COLUMN-(k-k_0+1+trim);

if(dk==0){
}/**if(dk)**/
else{
kmax[fn]+=dk;if(kmax[fn]>kceil[fn]) reallocflag=p_realloc();
if(reallocflag==0)
memmove(&p[fn][k+dk],&p[fn][k],kmax[fn]-dk-k+1);
}/**else(dk)**/
}/**if(p[fn][k],k)*******************************/
else{
if(gettype_p(k)==3) k++;

while(1){
member=top_icsr(/*firstline+*/jcsr,icsr);
type=gettype_p(member);
if(type<=2) {if(p[fn][member]!=' ' && p[fn][member]!=0x09) break;}
else if(type==3){
if(p[fn][member]!=/*0xa1*/SPC1 || p[fn][member+1]!=/*0xa1*/SPC2) break;
}
else{}

if(icsr==0) return;

csr_left();
}

k_right=member;if(gettype_p(k_right)==3) k_right++;

csr_row_home();
while(1){
member=top_icsr(/*firstline+*/jcsr,icsr);
```

```
type=gettype_p(member);
if(type<=2) {if(p[fn][member]!=' ' && p[fn][member]!=0x09) break;}
else if(type==3){
if(p[fn][member]!=/*0xa1*/SPC1 || p[fn][member+1]!=/*0xa1*/SPC2) break;
}
else{}

csr_right();
}

k_left=member;

if((length=while_puts_thepart(k_left,k_right))==COLUMN+1) return;
memmove(&buf_line[0],&p[fn][k_left],k_right-k_left+1);

k_0=top_icsr(/*firstline+*/jcsr,0);
trim=length-(k_right-k_left+1);
dk=COLUMN-(k-k_0+1+trim);

if(dk==0){
}/**if(dk)**/
else if(dk>0){
kmax[fn]+=dk;if(kmax[fn]>kceil[fn]) reallocflag=p_realloc();
if(reallocflag==0)
memmove(&p[fn][k+dk],&p[fn][k],kmax[fn]-dk-k+1);
}/**else if(dk)**/
else{
/*beep(50);*/
csr_row_end();
lumpflag=1;
deletion_onlymem();
uflag=1;
insertion(' ');
uflag=0;
lumpflag=0;
}/**else(dk)**/
}/**else(p[fn][k],k)**/

if(reallocflag==0){
member=k_0;
while(1){
p[fn][member]=' ';
if(member==k+dk) break;

member++;
}
```

```
if(flag==0)
dk_centering=0;
else if(flag==1)
dk_centering=(COLUMN-length)/2;
else
dk_centering=COLUMN-length;

memmove(&p[fn][k_0+dk_centering],&buf_line[0],k_right-k_left+1);

while_puts_show_(1,firstk);      /* 1 : TextOut to plane_1 */
k1=k_0+dk_centering+(k_right-k_left+1);
k2=top_icsr(jcsr+1,0);
dk_deletion=k2-k1;
if(dk_deletion>0){
memmove(&p[fn][k1],&p[fn][k2],kmax[fn]-k2+1);
kmax[fn]-=dk_deletion;

kmax[fn]+=1;
memmove(&p[fn][k1+1],&p[fn][k1],kmax[fn]-1-k1+1);
p[fn][k1]='\n';
}
}/**if(reallocflag)**/
else{
flag_=2;
kmax[fn]-=dk;
}/**else(reallocflag)**/

icsr=0;
page_firstk(firstk);

if(flag_==0){
if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;
}
}/** hcentering **/


void keydowns_f2(void)
{
if(GKS('Y')<0 || GKS('y')<0){
  charflag=0;charcode=0;}
else if(GKS('N')<0 || GKS('n')<0){
  charflag=0;charcode=1;}
else if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  charflag=0;charcode=2;}
else{
```

```
  charflag=0;charcode=3;}
}/** keydowns_f2 **/


void restore_in_PAINT(void)
{
/*restoreflag=1;*/
restore_3(1);
/*restoreflag=0;*/
}/** restore_in_PAINT **/


void breaks(char flag)
{
extraline(0);cqflag=0;
if(flag==1){
charflag=0;charcode=2;
}
}/** breaks **/


void imm_pause(void)
{
XUnsetICFocus(ic);
imm_restart_flag=1;
}/** imm_pause **/


void imm_restart(void)
{
XSetICValues(ic,XNFocusWindow,w,NULL);
XSetICFocus(ic);
InputPosition(ic,icsr,jcsr);
/*imm_restart_flag=0;*/                 /* no problem ? */
}/** imm_restart **/


void imm_restart_filer(void)
{
XSetICValues(ic,XNFocusWindow,w,NULL);
XSetICFocus(ic);
/*InputPosition(ic,icsr,jcsr);*/
/*imm_restart_flag=0;*/                 /* no problem ? */
}/** imm_restart_filer **/
```

```
void nest_free(void)
{
charflag=0;charcode=2;
BitBltflag=2;

nest_free_flag=1;
nest=0;
}/** nest_free **/


void fopen_successed(void)
{
fsp--;
if(fstack[fsp].flag==0) fn=fsp;
else fn=fstack[fsp].number;

ftable[ftp].fn=fn;
ftp++;
}/** fopen_successed **/


void fload_failed(void)
{
fstack[fsp].flag=1;
fstack[fsp].number=fn;
fsp++;

ftp--;
}/** fload_failed **/


long gethead_dialog(char flag,long member)
{
char type;
long k,dk_auto;

k=0;dk_auto=0;

while(1){
if(k==member) return k;
if(k>member){
if(flag==0) k-=dk_auto;else k=k;
return k;
}

type=gettype_dialog(k);
```

```
if(type<=2) {dk_auto=1;k+=dk_auto;}
else if(type==3) {dk_auto=2;k+=dk_auto;}
else{}
}
}/** gethead_dialog **/


int ishead_dialog(long member)
{
char type;
int dk_auto;
long k;

k=firstk_dialog;dk_auto=0;

while(1){
if(k==member) return 0;
if(k>member) return dk_auto;

type=gettype_dialog(k);

if(type<=2) {dk_auto=1;k+=dk_auto;}
else if(type==3) {dk_auto=2;k+=dk_auto;}
else{}
}
}/** ishead_dialog **/


int ishead_ac(long member)
{
char type;
int dk_auto;
long k;

k=0;dk_auto=0;

while(1){
if(k==member) return 0;
if(k>member) return dk_auto;

type=gettype_ac(k);

if(type<=2) {dk_auto=1;k+=dk_auto;}
else if(type==3) {dk_auto=2;k+=dk_auto;}
else{}
```

```
}
}/** ishead_ac **/


long get_firstk(long kend,long dline_)
{
long dline;                         /* dline, dline_ : auto */
long kstart;

kstart=get_kstart(kend);
dline=while_puts_dline(kstart,kend);

if(dline-dline_>0) firstk=while_puts_firstk(kstart,dline-dline_);
else if(dline-dline_==0) firstk=kstart;
else firstk=0;                      /* or kstart */

return dline-dline_;
}/** get_firstk **/


long get_kstart(long member)
{
char type;
long member_,k;

member=member-(COLUMN+1)*(ROW+1/*2*/);if(member<=1) return 0;  /* kstart : 0 */

member_=member;
while(1){
if(member_==0) return 0;            /* kstart : 0 */
if(p[fn][member_]=='\n') {/*member_++;*/break;}

member_--;
}

member=member_;

while(1){

member_=member;
while(1){
member_--;

if(member_==0) return 0;            /* kstart : 0 */
if(p[fn][member_]=='\n') {member_++;break;}
}
```

```
k=member_;
while(1){
if(k>member) break;
if(p[fn][k]=='\n') {k++;return k;}  /* kstart : k */

type=gettype_p(k);
if(type<=2) k+=1;
else if(type==3) k+=2;
else{}
}

member=member_-1;                   /* old '\n' */
}/**while(1)**/
}/** get_kstart **/


int ishead_(long member,long member_)
{
char type;
int dk_auto;
long /*member_,*/k;

/*member_=member;

while(1){
if(member_==0) break;
if(p[fn][member_]=='\n') {member_++;break;}

member_--;
}*/

k=member/*_*/;dk_auto=0;

while(1){
if(k==member+member_) return 0;
if(k>member+member_) return dk_auto;

type=gettype_p(k);

if(type<=2) {dk_auto=1;k+=dk_auto;}
else if(type==3) {dk_auto=2;k+=dk_auto;}
else{}
}
}/** ishead_ **/
```

```
int ishead(long member)
{
char type;
int dk_auto;
long member_,k;

member_=member;

while(1){
if(member_==0) break;
if(p[fn][member_]=='\n') {member_++;break;}

member_--;
}

k=member_;dk_auto=0;

while(1){
if(k==member) return 0;
if(k>member) return dk_auto;

type=gettype_p(k);

if(type<=2) {dk_auto=1;k+=dk_auto;}
else if(type==3) {dk_auto=2;k+=dk_auto;}
else{}
}
}/** ishead **/


void insertion_cc_dialog(unsigned char charcode)
{
unsigned char i;

if(charcode>=0x61 && charcode<=0x7a) charcode-=0x20;

/*if(dialogflag_REF==0 && charcode=='J') return;*/

i=0;
while(1){
if(charcode==cc[i]) break;

i++;
if(i==0x20) break;
}
```

```
if(i<0x20){
if(i==0x00) i=0x7f;
insertion_dialog(i);}
}/** insertion_cc_dialog **/


void insertion_cc(unsigned char charcode)
{
unsigned char i;

if(charcode>=0x61 && charcode<=0x7a) charcode-=0x20;

i=0;
while(1){
if(charcode==cc[i]) break;

i++;
if(i==0x20) break;
}

if(i<0x20){
if(i==0x00) i=0x7f;
insertion(i);}
}/** insertion_cc **/


void write_3vals(int fcp)
{
ftable[fcp].icsr=icsr;
ftable[fcp].jcsr=jcsr;
ftable[fcp].firstk=firstk;
}/** write_3vals **/


void read_3vals(int fcp)
{
icsr=ftable[fcp].icsr;
jcsr=ftable[fcp].jcsr;
firstk=ftable[fcp].firstk;

if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}
}/** read_3vals **/


void switch_division(char flag)
```

```
{
int fcp,csrcolor_tmp,i;

if(divideflag==0) return;
cut=0;

if(flag==0){
if(divisionnumber==2){
fn_2nd=fn;

fcp=0;
while(1){
if(ftable[fcp].fn==fn_1st) break;
fcp++;}

if(fcp<ftp-1){
ft_tmp=ftable[fcp];
/*memmove(&ftable[fcp],&ftable[fcp+1],sizeof(ft)*(ftp-1-fcp));*/
for(i=fcp;i<=ftp-2;i++) ftable[i]=ftable[i+1];
ftable[ftp-1]=ft_tmp;
}

DJ=ROW+2;
divisionnumber=2;
mlinecolor=1;

fn=fn_2nd;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
csrcolor_tmp=(csrcolor>1)?(csrcolor-1):15;
setcsrcolor(csrcolor_tmp);
csr();csr_to_1();
setcsrcolor(csrcolor);              /* restore */

DJ=0;
divisionnumber=1;
mlinecolor=0;

read_3vals(ftp-1);
fn=fn_1st;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
}
}/**if(flag)**/
else{
if(divisionnumber==1){
```

```
fn_1st=fn;

fcp=0;
while(1){
if(ftable[fcp].fn==fn_2nd) break;
fcp++;}

if(fcp<ftp-1){
ft_tmp=ftable[fcp];
/*memmove(&ftable[fcp],&ftable[fcp+1],sizeof(ft)*(ftp-1-fcp));*/
for(i=fcp;i<=ftp-2;i++) ftable[i]=ftable[i+1];
ftable[ftp-1]=ft_tmp;
}

DJ=0;
divisionnumber=1;
mlinecolor=1;

fn=fn_1st;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
csrcolor_tmp=(csrcolor>1)?(csrcolor-1):15;
setcsrcolor(csrcolor_tmp);
csr();csr_to_1();
setcsrcolor(csrcolor);              /* restore */

DJ=ROW+2;
divisionnumber=2;
mlinecolor=0;

read_3vals(ftp-1);
fn=fn_2nd;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
}
}/**else(flag)**/
}/** switch_division **/


void use_subroop(void)
{
char function_old,charflag_old;

usflag=1;

function_old=function;function=2;
```

```
charflag_old=charflag;
imm_pause();

yorn=subroop();

function=function_old;
charflag=charflag_old;
imm_restart();
}/** use_subroop **/


void divide_display(void)
{
char editflag_old,divideflag_old;
int fcp_1st,fcp_2nd,csrcolor_tmp,i;
unsigned char home[ASIZE];

if(ROW_L<10) return;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

divideflag_old=divideflag;
divideflag=1;

_1st:

divideflag_=1;

mnuproc_MULTIFILE("Divide");
cut=0;
if(refill==0){
divideflag=divideflag_old;divideflag_=0;refill=1;
page_firstk(firstk);
goto end;
}
fcp_1st=jcsr_select-1;

divideflag_=2;

mnuproc_MULTIFILE("Divide");
if(refill==0) {refill=1;goto _1st;}
fcp_2nd=jcsr_select-1;

divideflag_=0;

if(fcp_1st==fcp_2nd){
```

```
if(fsp<1 || editflag[ftable[fcp_1st].fn]<=-1){
divideflag=divideflag_old;
if(fsp<1) message(9,-1);else message(13,-1);
page_firstk(firstk);
goto end;
}

strcpy(home,home_global);
strcat(home,"zzz.copy");
strcpy(fname,home);
fn=ftable[fcp_1st].fn;
nobeepflag=1;
/*editflag_old=editflag[fn];*/
fsave(0,0);
/*editflag[fn]=editflag_old;*/
nobeepflag=0;

if(fopen_()==1){
divideflag=divideflag_old;
message(6,-1);
show_top(0);
goto end;
}

read_3vals(fcp_1st);

fopen_successed();
/*fsp--;
if(fstack[fsp].flag==0) fn=fsp;
else fn=fstack[fsp].number;

ftable[ftp].fn=fn;
ftp++;*/

/*icsr=0;jcsr=0;firstline=0;*/
lumpflag=1;
if(fload(0)==1){
divideflag=divideflag_old;lumpflag=0;
fload_failed();read_3vals(ftp-1);show_top(0);
goto end;
}
lumpflag=0;
unlink(fname);
strcpy(fname,fnames[ftable[fcp_1st].fn]);
strcpy(fnames[fn],fname);editflag[fn]=0;  /* copy filename */
```

```
fcp_1st=ftp-1;
write_3vals(fcp_1st);
}/**if(fcp_1st,fcp_2nd)**/
else{
if(fcp_1st<ftp-1){
ft_tmp=ftable[fcp_1st];
/*memmove(&ftable[fcp_1st],&ftable[fcp_1st+1],sizeof(ft)*(ftp-1-fcp_1st));*/
for(i=fcp_1st;i<=ftp-2;i++) ftable[i]=ftable[i+1];
ftable[ftp-1]=ft_tmp;
}

if(fcp_1st<fcp_2nd) fcp_2nd--;
fcp_1st=ftp-1;
}/**else(fcp_1st,fcp_2nd)**/

fn_1st=ftable[fcp_1st].fn;
fn_2nd=ftable[fcp_2nd].fn;

ROW=ROW_S;
DJ=ROW+2;
divisionnumber=2;
mlinecolor=1;

read_3vals(fcp_2nd);
/*icsr=0;jcsr=0;firstline=0;*/
fn=fn_2nd;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
csrcolor=CSRCOLOR;                    /* restore */
csrcolor_tmp=(csrcolor>1)?(csrcolor-1):15;
setcsrcolor(csrcolor_tmp);
csr();csr_to_1();

DJ=0;
divisionnumber=1;
mlinecolor=0;

read_3vals(fcp_1st);
/*icsr=0;jcsr=0;firstline=0;*/
fn=fn_1st;
strcpy(fname,fnames[fn]);
page_firstk(firstk);

end:
setcsrcolor((csrcolor=CSRCOLOR));
if(divideflag==1) restore_another();
```

```
/*999*/
if(1) use_subroop();
}/** divide_display **/


void restore_display(void)
{
if(divideflag==0) return;

divideflag=0;
ROW=ROW_L;DJ=0;
divisionnumber=0;

fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
}/** restore_display **/


void show_file(void)
{
char editflag_old;
int fcp,fn_tmp,i;
unsigned char home[ASIZE],str[20],buf[5];

setcsrcolor((csrcolor=CSRCOLOR_FILER));

/*itoa(FMAX-1,buf,10);*/gcvt(FMAX-1,5,buf);
sprintf(str,"Show(maxfiles=%s)",buf);

mnuproc_MULTIFILE(str);
cut=0;
if(refill==0) {refill=1;page_firstk(firstk);goto end;}
fcp=jcsr_select-1;
if(fcp==ftp-1) {page_firstk(firstk);goto end;}

if(divideflag==1){
fn_tmp=ftable[fcp].fn;
if((divisionnumber==1 && fn_tmp==fn_2nd) || (divisionnumber==2 && fn_tmp==fn_1st)){
if(fsp<1 || editflag[fn_tmp]<=-1){
if(fsp<1) message(9,-1);else message(13,-1);
page_firstk(firstk);
goto end;
}
```

```
strcpy(home,home_global);
strcat(home,"zzz.copy");
strcpy(fname,home);
fn=ftable[fcp].fn;
nobeepflag=1;
/*editflag_old=editflag[fn];*/
fsave(0,0);
/*editflag[fn]=editflag_old;*/
nobeepflag=0;

if(fopen_()==1){
message(6,-1);
show_top(0);
goto end;
}

read_3vals(fcp);

fopen_successed();
/*fsp--;
if(fstack[fsp].flag==0) fn=fsp;
else fn=fstack[fsp].number;

ftable[ftp].fn=fn;
ftp++;*/

/*icsr=0;jcsr=0;firstline=0;*/
if(fload(0)==1){
fload_failed();read_3vals(ftp-1);show_top(0);
goto end;
}
unlink(fname);
strcpy(fname,fnames[ftable[fcp].fn]);
strcpy(fnames[fn],fname);editflag[fn]=0;  /* copy filename */

write_3vals(ftp-1);
}/**if(divisionnumber,fn_tmp)**/
else{
read_3vals(fcp);

ft_tmp=ftable[fcp];
/*memmove(&ftable[fcp],&ftable[fcp+1],sizeof(ft)*(ftp-1-fcp));*/
for(i=fcp;i<=ftp-2;i++) ftable[i]=ftable[i+1];
ftable[ftp-1]=ft_tmp;

/*icsr=0;jcsr=0;firstline=0;*/
```

```
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
}/**else(divisionnumber,fn_tmp)**/
}/**if(divideflag)**/
else{
read_3vals(fcp);

ft_tmp=ftable[fcp];
/*memmove(&ftable[fcp],&ftable[fcp+1],sizeof(ft)*(ftp-1-fcp));*/
for(i=fcp;i<=ftp-2;i++) ftable[i]=ftable[i+1];
ftable[ftp-1]=ft_tmp;

/*icsr=0;jcsr=0;firstline=0;*/
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
}/**else(divideflag)**/

end:
setcsrcolor((csrcolor=CSRCOLOR));
if(divideflag==1) restore_another();

end_:{}
}/** show_file **/


void restore_another(void)
{
int fcp,csrcolor_tmp;

if(divisionnumber==1){
DJ=ROW+2;  /* lower */
divisionnumber=2;
mlinecolor=1;

fcp=0;
while(1){
if(ftable[fcp].fn==fn_2nd) break;
fcp++;}
read_3vals(fcp);

fn=fn_2nd;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
csrcolor_tmp=(csrcolor>1)?(csrcolor-1):15;
```

```
setcsrcolor(csrcolor_tmp);
csr();csr_to_1();
setcsrcolor(csrcolor);              /* restore */

DJ=0;
divisionnumber=1;
mlinecolor=0;

read_3vals(ftp-1);
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
/*page_firstk(firstk);*/
while_puts_show_(0,firstk);
}
else{
DJ=0;  /* upper */
divisionnumber=1;
mlinecolor=1;

fcp=0;
while(1){
if(ftable[fcp].fn==fn_1st) break;
fcp++;}
read_3vals(fcp);

fn=fn_1st;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
csrcolor_tmp=(csrcolor>1)?(csrcolor-1):15;
setcsrcolor(csrcolor_tmp);
csr();csr_to_1();
setcsrcolor(csrcolor);              /* restore */

DJ=ROW+2;
divisionnumber=2;
mlinecolor=0;

read_3vals(ftp-1);
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
/*page_firstk(firstk);*/
while_puts_show_(0,firstk);
}
}/** restore_another **/
```

```
void show_top(char flag)
{
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
if(flag==1) csr();
}/** show_top **/


void save_all(char flag)
{
/*char flag_saved=0;*/
int j;

j=ftp;
while(1){
fn=ftable[j-1].fn;

if(editflag[fn]==1){
read_3vals(j-1);
strcpy(fname,fnames[fn]);
/*if(*/fsave(1,0)/*==0 && flag_saved==0) flag_saved=1*/;
}

j--;
if(j<1) break;
}

if(flag==0){
read_3vals(ftp-1);
show_top(/*0*/1);                    /* verbose ! */

/*if(flag_saved) beep(50);*/
puts_mline(0,"All saved.");
noelineflag=1;
}
else{
end_ble();
}
}/** save_all **/


void close_all(void)
{
int j;
/*int icsr_old,jcsr_old;
```

```
long firstline_old;*/

nocloseflag=0;
passflag=0;

/*firstline_old=firstline;
icsr_old=icsr;jcsr_old=jcsr;*/

j=ftp;
while(1){
fn=ftable[j-1].fn;

if(editflag[fn]==1){
/*if(j<ftp) */read_3vals(j-1);
beep(50);delay_(100);beep(50);
dlgproc_SAVE_();/*delay_(200);*/
}

if(nocloseflag==1){
/*firstline=firstline_old;
icsr=icsr_old;jcsr=jcsr_old;*/
read_3vals(ftp-1);
show_top(0);
return;
}

j--;
if(j<1) break;
}

if(passflag==1){
/*firstline=firstline_old;
icsr=icsr_old;jcsr=jcsr_old;*/
read_3vals(ftp-1);
show_top(/*0*/1);

message(4,1);
if(yorn!=0) return;
}

end_ble();
}/** close_all **/


void end_ble(void)
{
```

```
int fn_tmp;

if(ftp>0){
while(1){
fn_tmp=ftable[ftp-1].fn;
free(p[fn_tmp]);/*free(ptmp);*/

ftp--;
if(ftp<1) break;
}
}

/*refill=0;charflag=0;charcode=2;*/
closegraph_();exit(0);
}/** end_ble **/


void close_open(char saveflag)
{
int j;
int fn_tmp;
unsigned char oldstring[ASIZE];

/* close_all() */
nocloseflag=0;
passflag=0;

/*firstline_old=firstline;
icsr_old=icsr;jcsr_old=jcsr;*/

if(saveflag){
j=ftp;
while(1){
fn=ftable[j-1].fn;

if(editflag[fn]==1/* && saveflag==1*/){
/*if(j<ftp) */read_3vals(j-1);
beep(50);delay_(100);beep(50);
dlgproc_SAVE_();/*delay_(200);*/
}

if(nocloseflag==1){
/*firstline=firstline_old;
icsr=icsr_old;jcsr=jcsr_old;*/
read_3vals(ftp-1);
show_top(0);
```

```
return;
}

j--;
if(j<1) break;
}
}

if(passflag==1){
/*firstline=firstline_old;
icsr=icsr_old;jcsr=jcsr_old;*/
read_3vals(ftp-1);
show_top(/*0*/1);

message(/*4*/3,1);
if(yorn!=0) return;
}

/*end_ble();*/                        /* no closegraph_();exit(0); */
if(ftp>0){
while(1){
fn_tmp=ftable[ftp-1].fn;
free(p[fn_tmp]);/*free(ptmp);*/

ftp--;
if(ftp<1) break;
}
}



/*9*/
ftp=0;
fsp=FMAX-1-ftp;
fn=0;



/* part of close_file() */
if(ftp<1){
divideflag=0;
ROW=ROW_L;DJ=0;
divisionnumber=0;

cleardevice_(-1,0,0,0,0);
BitBlt_nomline();
```

```
extraline(0);

/* unit -> */
strcpy(oldstring,fname);
start:

dlgproc_OPEN(0);
if(refill==0) {closegraph_();exit(0);}
if(fopen_()==1){
strcpy(fname,/*fname_old*/oldstring);
/*message(1,1);*/puts_mline(0,"Reinput a filename.");goto start;}

fopen_successed();

/*strcpy(fnames[fn],fname);editflag[fn]=0;*/
icsr=0;jcsr=0;firstk=0;
if(fload(0)==1){
fload_failed();/*fname[0]='\0';*/strcpy(fname,/*fname_old*/oldstring);goto start;}
strcpy(fnames[fn],fname); editflag[fn]=0;
sizeoffname=max(strlen(fname),sizeoffname);
if(unlinkflag) unlink(fname);
/* <- unit */
}/**if(ftp)**/
}/** close_open **/


void close_file(void)
{
char displayflag;
int fcp;
unsigned char oldstring[ASIZE];

nocloseflag=0;
passflag=0;

/*fcp=ftp-1;
fn=ftable[fcp].fn;*/

if(editflag[fn]==1){
beep(50);delay_(100);beep(50);
dlgproc_SAVE(0);
}

if(nocloseflag==1) return;

if(passflag==1){
```

```
csr();

message(5,1);
if(yorn!=0) return;
}

free(p[fn]);/*free(ptmp);*/
fload_failed();
/*fstack[fsp].flag=1;
fstack[fsp].number=fn;
fsp++;

ftp--;*/

if(ftp<1){
cleardevice_(-1,0,0,0,0);
BitBlt_nomline();
extraline(0);

/* unit -> */
strcpy(oldstring,fname);
start:

dlgproc_OPEN(0);
if(refill==0) {closegraph_();exit(0);}
if(fopen_()==1){
strcpy(fname,/*fname_old*/oldstring);
/*message(1,1);*/puts_mline(0,"Reinput a filename.");goto start;}

fopen_successed();

/*strcpy(fnames[fn],fname);editflag[fn]=0;*/
icsr=0;jcsr=0;firstk=0;
if(fload(0)==1){
fload_failed();/*fname[0]='\0';*/strcpy(fname,/*fname_old*/oldstring);goto start;}
strcpy(fnames[fn],fname); editflag[fn]=0;
sizeoffname=max(strlen(fname),sizeoffname);
if(unlinkflag) unlink(fname);
/* <- unit */
}/**if(ftp)**/
else{
if(divideflag==1){
displayflag=1;

fcp=ftp-1;fn=ftable[fcp].fn;
if(divisionnumber==1) fn_1st=fn;
```

```
else fn_2nd=fn;

if(fn_1st==fn_2nd){
if(fcp==0){
read_3vals(fcp);
displayflag=0;restore_display();
}
else{
ft_tmp=ftable[fcp];
ftable[fcp]=ftable[fcp-1];
ftable[fcp-1]=ft_tmp;
}
}/**if(fn_1st,fn_2nd)**/

if(displayflag==1){
read_3vals(ftp-1);

/*icsr=0;jcsr=0;firstline=0;*/
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
}
}/**if(divideflag)**/
else{
read_3vals(ftp-1);

/*icsr=0;jcsr=0;firstline=0;*/
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
}/**else(divideflag)**/
}/**else(ftp)**/
}/** close_file **/


void open_file(char flag_rn)
{
unsigned char oldstring[ASIZE];

if(fsp<1){
message(9,1);csr();
return;
}

if(flag_rn==1) {puts_mline(0,"Read-only file");noelineflag=1;}
else if(flag_rn==2) {puts_mline(0,"New file");noelineflag=1;}
```

```
/* unit -> */
strcpy(oldstring,fname);
start:

dlgproc_OPEN(flag_rn);                      /* firstk_old, icsr_old, jcsr_old */
if(refill==0){
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
page_firstk(firstk);
refill=1;return;}
if(flag_rn==2) newopen=1;else newopen=0;
if(fopen_()==1){
strcpy(fname,/*fname_old*/oldstring);
/*message(1,1);*/puts_mline(0,"Reinput a filename.");goto start;}

fopen_successed();

icsr=0;jcsr=0;firstk=0;
if(fload(flag_rn)==1){              /* flag_rn */
fload_failed();/*fname[0]='\0';*/strcpy(fname,/*fname_old*/oldstring);goto start;}
strcpy(fnames[fn],fname);
sizeoffname=max(strlen(fname),sizeoffname);
if(unlinkflag) unlink(fname);
/* <- unit */
}/** open_file **/


void file_attri(void)
{
if(editflag[fn]==-1) {editflag[fn]=0;/*strcpy(attri,"");*/}
else if(editflag[fn]==-2) {editflag[fn]=1;/*strcpy(attri,"");*/}
else if(editflag[fn]==0) {editflag[fn]=-1;/*strcpy(attri,"RO  ");*/}
else if(editflag[fn]==1) {editflag[fn]=-2;/*strcpy(attri,"RO  ");*/}
else ;
}/** file_attri **/


void copy_cfg(void)
{
unsigned char src[ASIZE],dst[ASIZE];

strcpy(src,home_global);
strcat(src,"org_sj.cfg");
strcpy(dst,home_global);
strcat(dst,"ble.cfg");
```

```
if(CopyFile(src,dst)!=-1)
puts_mline(0,"Default setup");
}/** copy_cfg **/


int CopyFile(unsigned char *src,unsigned char *dst)
{
int val;
unsigned char cmd[ASIZE];

strcpy(cmd,"cp -fp ");
strcat(cmd,src);
strcat(cmd," ");
strcat(cmd,dst);
val=system(cmd);

return val;
}/** CopyFile **/


void edit_tmpcfg(void)
{
char flag_open;
int dm;
long member,member_,member_RETURN;
unsigned char home[ASIZE];

fn=FMAX-1;
refflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

strcpy(home,home_global);
strcat(home,"bletmp.cfg");

strcpy(fname,home);

if((flag_open=fopen_())==1) message(6,0);
firstk=firstk_cfg;                  /* share 3vals */
icsr=icsr_cfg;jcsr=jcsr_cfg;
/*lumpflag=1;*/
if(flag_open==1 || fload(0)==1){
strcpy(array,"");
goto end_;}
```

```
within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}  /* from read_3vals() */

/*lumpflag=0;
text_end();
csr_row_home();csr_up();monitorline(1);*/
csr();

mainroop();

if(refill!=-1 && editflag[fn]==1){
if(fsize_g==0) {unlink(home);goto end_;}
fsave(0,0);

if((fp=fopen(home,"rb"))!=NULL){
fseek(fp,0L,SEEK_END);
cfgmax=ftell(fp)-1;
pcfg=(unsigned char *)malloc((cfgmax+1)+(0+1)*sizeof(unsigned char));
if(pcfg!=NULL){
fseek(fp,0L,SEEK_SET);
fread(pcfg,1,(cfgmax+1),fp);
fclose(fp);

if((l_s_flag=read_cfg(20))==-1000) l_s_flag=0;
else l_s_flag=min(l_s_flag,1);

if((tabspaces=read_cfg(21))==-1000) tabspaces=0;
else tabspaces=min(tabspaces,1);

free(pcfg);
}
else fclose(fp);
}/**if(fp)**/
}
free(p[fn]);/*free(ptmp);*/

firstk_cfg=firstk;
icsr_cfg=icsr;jcsr_cfg=jcsr;

end_:
setcsrcolor((csrcolor=CSRCOLOR));
refflag=0;
refill=1;
}/** edit_tmpcfg **/
```

```
void edit_cfg(void)
{
char flag_open;
int dm;
long member,member_,member_RETURN;
unsigned char buf[ASIZE],home[ASIZE];

fn=FMAX-1;
refflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

strcpy(home,home_global);
strcat(home,"ble.cfg");

strcpy(fname,home);

if((flag_open=fopen_())==1) message(6,0);
firstk=firstk_cfg;
icsr=icsr_cfg;jcsr=jcsr_cfg;
/*lumpflag=1;*/
if(flag_open==1 || fload(0)==1){
strcpy(array,"");
goto end_;}

within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}  /* from read_3vals() */

/*lumpflag=0;
text_end();
csr_row_home();csr_up();monitorline(1);*/
csr();

mainroop();

if(refill!=-1) fsave(0,0);
free(p[fn]);/*free(ptmp);*/

firstk_cfg=firstk;
icsr_cfg=icsr;jcsr_cfg=jcsr;

end_:
setcsrcolor((csrcolor=CSRCOLOR));
refflag=0;
refill=1;
}/** edit_cfg **/
```

```
void string_visible(void)
{
char flag_open;
int dm;
long member,member_,member_RETURN;
unsigned char buf[ASIZE];

member=topp[/*firstline+*/jcsr]+0;
while(1){
if(member==kmax[fn]) goto end;
/*if(member==kmax[fn]) break;*/
if(p[fn][member]=='\n' && ishead(member)==0) break;

member++;
}

member_RETURN=member;

if(member>0) member--;
if(p[fn][member]=='\n' && ishead(member)==0) goto end;
member_=member;
while(1){
if(member_==0) break;
if(p[fn][member_]=='\n' && ishead(member_)==0) {member_++;break;}

member_--;
}

dm=member_RETURN-member_;
if(dm>ASIZEM-1) dm=0;

strncpy(buf,&p[fn][member_],dm);
buf[dm]='\0';

strcpy(array,buf);
goto end_;

end:
strcpy(array,"");

end_:{}
}/** string_visible **/
```

```
void copy_string(void)
{
char flag_open;
int dm;
long member,member_,member_RETURN;
unsigned char buf[ASIZE],home[ASIZE];

fn=FMAX-1;
refflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

strcpy(home,home_global);
strcat(home,"zzz.string");

strcpy(fname,home);

if((flag_open=fopen_())==1) message(6,0);
editflag[fn]=0;
firstk=firstk_string;
icsr=icsr_string;jcsr=jcsr_string;
/*lumpflag=1;*/
if(flag_open==1 || fload(0)==1){
strcpy(array,"");
goto end_;}

within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}  /* from read_3vals() */

/*lumpflag=0;
text_end();
csr_row_home();csr_up();monitorline(1);*/
csr();

mainroop();

if(refill==-2){
member=topp[/*firstline+*/jcsr]+0;
while(1){
if(member==kmax[fn]) goto end;
/*if(member==kmax[fn]) break;*/
if(p[fn][member]=='\n' && ishead(member)==0) break;

member++;
}
```

```
member_RETURN=member;

if(member>0) member--;
if(p[fn][member]=='\n' && ishead(member)==0) goto end;
member_=member;
while(1){
if(member_==0) break;
if(p[fn][member_]=='\n' && ishead(member_)==0) {member_++;break;}

member_--;
}

dm=member_RETURN-member_;
if(dm>ASIZEM-1) dm=0;

strncpy(buf,&p[fn][member_],dm);
buf[dm]='\0';

strcpy(array,buf);
}/**if(refill)**/
else{
end:
strcpy(array,"");
}/**else(refill)**/

if(refill!=-1 && editflag[fn]==1) fsave(0,0);
free(p[fn]);/*free(ptmp);*/

firstk_string=firstk;
icsr_string=icsr;jcsr_string=jcsr;

end_:
setcsrcolor((csrcolor=CSRCOLOR));
refflag=0;
refill=1;
}/** copy_string **/


void program(void)
{
char flag_open;
int dm;
long member,member_,member_RETURN;
unsigned char buf[ASIZE],home[ASIZE];

fn=FMAX-1;
```

```
refflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

strcpy(home,home_global);
strcat(home,"zzz.program");

strcpy(fname,home);

if((flag_open=fopen_())==1) message(6,0);
editflag[fn]=0;
firstk=firstk_program;
icsr=icsr_program;jcsr=jcsr_program;
/*lumpflag=1;*/
if(flag_open==1 || fload(0)==1){
strcpy(array,"");
goto end_;}

within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}  /* from read_3vals() */

/*lumpflag=0;
text_end();
csr_row_home();csr_up();monitorline(1);*/
csr();

mainroop();

if(refill==-2){
member=topp[/*firstline+*/jcsr]+0;
while(1){
if(member==kmax[fn]) goto end;
/*if(member==kmax[fn]) break;*/
if(p[fn][member]=='\n' && ishead(member)==0) break;

member++;
}

member_RETURN=member;

if(member>0) member--;
if(p[fn][member]=='\n' && ishead(member)==0) goto end;
member_=member;
while(1){
if(member_==0) break;
if(p[fn][member_]=='\n' && ishead(member_)==0) {member_++;break;}
```

```
member_--;
}

dm=member_RETURN-member_;
if(dm>ASIZEM-1) dm=0;

strncpy(buf,&p[fn][member_],dm);
buf[dm]='\0';

strcpy(array,buf);
}/**if(refill)**/
else{
end:
strcpy(array,"");
}/**else(refill)**/

if(refill!=-1 && editflag[fn]==1) fsave(0,0);
free(p[fn]);/*free(ptmp);*/

firstk_program=firstk;
icsr_program=icsr;jcsr_program=jcsr;

end_:
setcsrcolor((csrcolor=CSRCOLOR));
refflag=0;
refill=1;
}/** program **/


void jump(void)
{
char flag_open;
int dm;
long member,member_,member_RETURN;
unsigned char buf[ASIZE],home[ASIZE];

fn=FMAX-1;
refflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

strcpy(home,home_global);
strcat(home,"zzz.jump");

strcpy(fname,home);
```

```
if((flag_open=fopen_())==1) message(6,0);
editflag[fn]=0;
firstk=firstk_jump;
icsr=icsr_jump;jcsr=jcsr_jump;
/*lumpflag=1;*/
if(flag_open==1 || fload(0)==1){
strcpy(array,"");
goto end_;}

within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}  /* from read_3vals() */

/*lumpflag=0;
text_end();
csr_row_home();csr_up();monitorline(1);*/
csr();

mainroop();

if(refill==-2){
member=topp[/*firstline+*/jcsr]+0;
while(1){
if(member==kmax[fn]) goto end;
/*if(member==kmax[fn]) break;*/
if(p[fn][member]=='\n' && ishead(member)==0) break;

member++;
}

member_RETURN=member;

if(member>0) member--;
if(p[fn][member]=='\n' && ishead(member)==0) goto end;
member_=member;
while(1){
if(member_==0) break;
if(p[fn][member_]=='\n' && ishead(member_)==0) {member_++;break;}

member_--;
}

dm=member_RETURN-member_;
if(dm>ASIZEM-1) dm=0;

strncpy(buf,&p[fn][member_],dm);
```

```
buf[dm]='\0';

strcpy(array,buf);
}/**if(refill)**/
else{
end:
strcpy(array,"");
}/**else(refill)**/

if(refill!=-1 && editflag[fn]==1) fsave(0,0);
free(p[fn]);/*free(ptmp);*/

firstk_jump=firstk;
icsr_jump=icsr;jcsr_jump=jcsr;

end_:
setcsrcolor((csrcolor=CSRCOLOR));
refflag=0;
refill=1;
}/** jump **/


void filename(void)
{
char flag_open;
int dm;
long member,member_,member_RETURN;
unsigned char buf[ASIZE],home[ASIZE];

fn=FMAX-1;
refflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

strcpy(home,home_global);
strcat(home,"zzz.filename");

strcpy(fname,home);

if((flag_open=fopen_())==1) message(6,0);
editflag[fn]=0;
firstk=firstk_filename;
icsr=icsr_filename;jcsr=jcsr_filename;
/*lumpflag=1;*/
if(flag_open==1 || fload(0)==1){
strcpy(array,"");
```

```
goto end_;}

within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}  /* from read_3vals() */

/*lumpflag=0;
text_end();
csr_row_home();csr_up();monitorline(1);*/
csr();

mainroop();

if(refill==-2){
member=topp[/*firstline+*/jcsr]+0;
while(1){
if(member==kmax[fn]) goto end;
/*if(member==kmax[fn]) break;*/
if(p[fn][member]=='\n' && ishead(member)==0) break;

member++;
}

member_RETURN=member;

if(member>0) member--;
if(p[fn][member]=='\n' && ishead(member)==0) goto end;
member_=member;
while(1){
if(member_==0) break;
if(p[fn][member_]=='\n' && ishead(member_)==0) {member_++;break;}

member_--;
}

dm=member_RETURN-member_;
if(dm>ASIZEM-1) dm=0;

strncpy(buf,&p[fn][member_],dm);
buf[dm]='\0';

strcpy(array,buf);
}/**if(refill)**/
else{
end:
strcpy(array,"");
}/**else(refill)**/
```

```
if(refill!=-1 && editflag[fn]==1) fsave(0,0);
free(p[fn]);/*free(ptmp);*/

firstk_filename=firstk;
icsr_filename=icsr;jcsr_filename=jcsr;

end_:
setcsrcolor((csrcolor=CSRCOLOR));
refflag=0;
refill=1;
}/** filename **/


void ref(void)
{
char flag_open;
int dm;
long member,member_,member_RETURN;
long stop;
unsigned char buf[ASIZE];

fn=FMAX-1;
refflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

strcpy(fname,home_ref);

if((flag_open=fopen_())==1) message(6,0);
editflag[fn]=0;
firstk=firstk_ref;
icsr=icsr_ref;jcsr=jcsr_ref;
/*lumpflag=1;*/
if(flag_open==1 || fload(0)==1){
strcpy(array,"");
goto end_;}

within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}  /* from read_3vals() */

/*lumpflag=0;
text_end();
csr_row_home();csr_up();monitorline(1);*/
csr();
```

```
mainroop();

if(refill==-2){                     /* Enter */
member=topp[/*firstline+*/jcsr]+0;
stop=member;

while(1){
if(member==kmax[fn]) goto end;
/*if(member==kmax[fn]) break;*/
if(p[fn][member]=='\n' && ishead(member)==0) break;

member++;
}


member_RETURN=member;

if(member==stop) goto end;          /* added */
if(member>0) member--;
if(p[fn][member]=='\n' && ishead(member)==0) goto end;
member_=member;

while(1){
if(member_==stop) break;            /* added */
if(member_==0) break;
if(p[fn][member_]=='\n' && ishead(member_)==0) {member_++;break;}

member_--;
}


dm=member_RETURN-member_;
if(dm>ASIZEM-1) dm=0;

strncpy(buf,&p[fn][member_],dm);
buf[dm]='\0';

strcpy(array,buf);
}/**if(refill)**/
else{                               /* Esc, Shift + Esc */
end:
strcpy(array,"");
}/**else(refill)**/

if(refill!=-1 && editflag[fn]==1) fsave(0,0);
free(p[fn]);/*free(ptmp);*/
```

```
firstk_ref=firstk;
icsr_ref=icsr;jcsr_ref=jcsr;

end_:
setcsrcolor((csrcolor=CSRCOLOR));
refflag=0;
refill=1;
}/** ref **/


void deleted(void)
{
char flag_open;

fn=FMAX-1;
deletedflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

strcpy(fname,home_deleted);

if((flag_open=fopen_())==1) message(6,0);
icsr=0;jcsr=0;firstk=0;
lumpflag=1;
if(flag_open==1 || fload(0)==1){
lumpflag=0;
goto end_;}

lumpflag=0;
text_end();
csr_row_home();csr_up();monitorline(1);
csr();

mainroop();

free(p[fn]);/*free(ptmp);*/

end_:
setcsrcolor((csrcolor=CSRCOLOR));
deletedflag=0;
refill=1;
}/** deleted **/


void filer(void)
```

```
{
if(T_TT==0) filer_t();
else filer_tt();
}/** filer **/


void filer_t(void)
{
char function_old;
char flag_,flag_open,flag_list;
int dm,length,len;
long k,dk_auto;
long member,member_,member_Return,member_Start;
unsigned char buf[ASIZE],buf_[ASIZE],home[ASIZE],cmd[ASIZE];

fn=FMAX-1;
filerflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

/*if(arraycheckflag==3) {chdir(array);cdflag=0;}*/

strcpy(home,home_global);
strcat(home,"zzz.filer");
strcpy(cmd,"ls -alo > ");
strcat(cmd,home);

while(1){
refill_old=-2;
driveflag=0;

unlink(home);
system(cmd);
strcpy(fname,home);

if((flag_open=fopen_())==1) message(6,/*0*/1);
icsr=0;jcsr=0;firstk=0;
lumpflag=2;
if(flag_open==1 || fload(0)==1){                    /* output a message */
lumpflag=0;
strcpy(array,"");
goto end_;}

getcwd(buf_,ASIZE);
length=strlen(buf_);
```

```
dk_auto=length;
strcat(buf_,"\n");dk_auto++;

k=0;
function_old=function;function=2;
member_last=k+dk_auto;                  /* or k+dk_auto */
flag_=pdata_increase(k,&buf_[0],dk_auto);
function=function_old;

if(flag_==0){
function_old=function;function=2;
jcsr=while_puts_dline(firstk,k+dk_auto);  /* or k+dk_auto */
function=function_old;
icsr=icsr_last;

jcsr+=DJCSR;
/*jcsr+=2;*/
/*jcsr++;*/                              /* tt.c */
/*;*/
jcsr_floor=jcsr;if(jcsr_floor/*+1*/>ROW-1) jcsr_floor=ROW-1/*-1*/;

while_puts_show_(0,firstk);      /* firstk : 0 */
topp_floor=topp[jcsr_floor/*+1*/];
}
else{
lumpflag=0;
strcpy(array,"");
free(p[fn]);/*free(ptmp);*/
goto end_;
}

if(cdflag==0){
firstk=0;
icsr=0;jcsr=jcsr_floor;
}
else{
firstk=firstk_filer;
icsr=/*0*/icsr_filer;jcsr=jcsr_filer;
}

within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}  /* from read_3vals() */

lumpflag=0;
page_firstk(firstk);
csr();
```

```
if(filerskip==1) nest_free_flag=1;

mainroop();

if(driveflag) goto end;

if(refill<=-2){                      /* Enter, Shift+Enter */
member=topp[/*firstline+*/jcsr]+0;

while(1){
if(member==kmax[fn]) goto end;
if(p[fn][member]=='\n') break;

member++;
}

member_Return=member;
if(member>0) member--;
if(p[fn][member]=='\n') goto end;
member_=member;

while(1){
if(member_==0) goto end;
if(p[fn][member_]=='\n') {member_++;break;}

member_--;
}

member_Start=member_;

while(1){
if(member==0) goto end;
if(p[fn][member]==' ' &&
   ((spacecheck(member_Start,member)==SPCNUM && spaces==1) ||
    p[fn][member-1]=='>')) break;

member--;
}

dm=member_Return-member-1;

strncpy(buf,&p[fn][member+1],dm);
buf[dm]='\0';

if(filer_execute){
```

```
filer_execute=0;
if(filerskip==1) filer_execute_phantom=1;

/*getcwd(buf_,ASIZE);*/buf_[length]='\0';  /* restore */
if(buf_[/*3*/1]!='\0') strcat(buf_,"/");
strcat(buf_,buf);

len=strlen(buf_);
if(len<ASIZEM-1-1){                 /* for string with spaces */
memmove(&buf_[1],buf_,len);
buf_[0]='\"';buf_[len+1]='\"';buf_[len+2]='\0';
}

strcpy(array,buf_);

systemflag=0;
if(strlen(array)!=0) execute(array);
firstk_filer=firstk;icsr_filer=icsr;jcsr_filer=jcsr;cdflag=-1;
goto end_execute;
}/**if(filer_execute)**/

/*if(access(buf,4)!=0) beep(1000);*/

                                            /* cdflag = -1, file, readable */
if((cdflag=chdir(buf))!=0 && p[fn][member_Start]!='d' &&
   p[fn][member_Start]!='p' && access(buf,4)==0){
/*getcwd(buf_,ASIZE);*/buf_[length]='\0';  /* restore */
if(buf_[/*3*/1]!='\0') strcat(buf_,"/");
if(p[fn][member_Start]=='l' && buf[0]=='/')
strcpy(buf_,buf);
else
strcat(buf_,buf);

strcpy(array,buf_);

if(refill==-2){
firstk_filer=firstk;icsr_filer=icsr;jcsr_filer=jcsr;/*cdflag=-1;*/  /* Enter */
}
else cdflag=0;                                          /* Shift + Enter */
refill_old=refill;
break;
}/**if(chdir(),dirflag,access())**/

end:
cdflag=0;
```

```
end_execute:
free(p[fn]);/*free(ptmp);*/
refill=1;
}/**if(refill)**/
else{
strcpy(array,"");
if(refill==0){
firstk_filer=firstk;icsr_filer=icsr;jcsr_filer=jcsr;cdflag=-1;  /* Esc */
}
else cdflag=0;                                          /* Shift + Esc */
refill_old=refill;
break;
}/**else(refill)**/
}/**while(1)**/

free(p[fn]);/*free(ptmp);*/
if(cdflag==0) chdir(home_global_GCD);
/*if(cdflag==0) SetCurrentDirectory(home_global_GCD);*/

end_:
setcsrcolor((csrcolor=CSRCOLOR));
filerflag=0;
refill=1;
}/** filer_t **/


void filer_tt(void)
{
char function_old;
char flag_,flag_open,flag_list;
int dm,length,len;
long k,dk_auto;
long member,member_,member_Return,member_Start;
unsigned char buf[ASIZE],buf_[ASIZE],home[ASIZE],/*cmd[ASIZE]*/count[64];

fn=FMAX-1;
filerflag=1;

setcsrcolor((csrcolor=CSRCOLOR_FILER));

/*if(arraycheckflag==3) {chdir(array);cdflag=0;}*/

strcpy(home,home_global);
strcat(home,"zzz.filer");
/*strcpy(cmd,"ls -alo > ");
strcat(cmd,home);*/
```

```
while(1){
refill_old=-2;
driveflag=0;

unlink(home);
/*system(cmd);*/
flag_list=make_list(home);
if(flag_list==1){
message(7,/*0*/1);
strcpy(array,"");
goto end_;}
if(flag_list==2){
chdir(home_global_GCD);
message(11,/*0*/1);
strcpy(array,"");
goto end_;}
strcpy(fname,home);

if((flag_open=fopen_())==1) message(6,/*0*/1);
icsr=0;jcsr=0;firstk=0;
lumpflag=2;
if(flag_open==1 || fload(0)==1){                    /* output a message */
lumpflag=0;
strcpy(array,"");
goto end_;}

getcwd(buf_,ASIZE);
length=strlen(buf_);

dk_auto=length;
strcat(buf_,"\n");dk_auto++;
sprintf(count,"Dir(s):%ld  File(s):%ld\n",count_dir,count_file);
strcat(buf_,count);dk_auto+=strlen(count);

k=0;
function_old=function;function=2;
member_last=k+dk_auto;                  /* or k+dk_auto */
flag_=pdata_increase(k,&buf_[0],dk_auto);
function=function_old;

if(flag_==0){
function_old=function;function=2;
jcsr=while_puts_dline(firstk,k+dk_auto);  /* or k+dk_auto */
function=function_old;
icsr=icsr_last;
```

```
jcsr+=DJCSR;
/*jcsr+=2;*/
/*jcsr++;*/                              /* tt.c */
/*;*/
jcsr_floor=jcsr;if(jcsr_floor/*+1*/>ROW-1) jcsr_floor=ROW-1/*-1*/;

while_puts_show_(0,firstk);      /* firstk : 0 */
topp_floor=topp[jcsr_floor/*+1*/];
}
else{
lumpflag=0;
strcpy(array,"");
free(p[fn]);/*free(ptmp);*/
goto end_;
}

if(cdflag==0){
firstk=0;
icsr=0;jcsr=jcsr_floor;
}
else{
firstk=firstk_filer;
icsr=/*0*/icsr_filer;jcsr=jcsr_filer;
}

within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;/*scroll_down(0);*/}  /* from read_3vals() */

lumpflag=0;
page_firstk(firstk);
csr();

if(filerskip==1) nest_free_flag=1;

mainroop();

if(driveflag) goto end;

if(refill<=-2){                      /* Enter, Shift+Enter */
member=topp[/*firstline+*/jcsr]+0;
if(/*p[fn][member]=='0'*/0) {beep(100);goto end;}    /* ex. 0, 0 */

while(1){
if(member==kmax[fn]) goto end;
if(p[fn][member]=='\n') break;
```

```
member++;
}

member_Return=member;
if(member>0) member--;
if(p[fn][member]=='\n') goto end;
member_=member;

while(1){
if(member_==0) goto end;
if(p[fn][member_]=='\n') {member_++;break;}

member_--;
}

member_Start=member_;

while(1){
if(member==0) goto end;
if(p[fn][member]==' ' &&
   ((spacecheck(member_Start,member)==SPCNUM && spaces==1) ||
    p[fn][member-1]=='>')) break;

member--;
}

dm=member_Return-member-1;

strncpy(buf,&p[fn][member+1],dm);
buf[dm]='\0';

if(filer_execute){
filer_execute=0;
if(filerskip==1) filer_execute_phantom=1;

/*getcwd(buf_,ASIZE);*/buf_[length]='\0';  /* restore */
if(buf_[/*3*/1]!='\0') strcat(buf_,"/");
strcat(buf_,buf);

len=strlen(buf_);
if(len<ASIZEM-1-1){                 /* for string with spaces */
memmove(&buf_[1],buf_,len);
buf_[0]='\"';buf_[len+1]='\"';buf_[len+2]='\0';
}
```

```
strcpy(array,buf_);

systemflag=0;
if(strlen(array)!=0) execute(array);
firstk_filer=firstk;icsr_filer=icsr;jcsr_filer=jcsr;cdflag=-1;
goto end_execute;
}/**if(filer_execute)**/

/*if(access(buf,4)!=0) beep(1000);*/

                                                    /* cdflag = -1, file, readable */
if((cdflag=chdir(buf))!=0 && p[fn][member_Start]!='d' &&
   p[fn][member_Start]!='p' && access(buf,4)==0){
/*getcwd(buf_,ASIZE);*/buf_[length]='\0';  /* restore */
if(buf_[/*3*/1]!='\0') strcat(buf_,"/");
if(p[fn][member_Start]=='l' && buf[0]=='/')
strcpy(buf_,buf);
else
strcat(buf_,buf);

strcpy(array,buf_);

if(refill==-2){
firstk_filer=firstk;icsr_filer=icsr;jcsr_filer=jcsr;/*cdflag=-1;*/  /* Enter */
}
else cdflag=0;                                              /* Shift + Enter */
refill_old=refill;
break;
}/**if(chdir(),dirflag,access())**/

end:
cdflag=0;

end_execute:
free(p[fn]);/*free(ptmp);*/
refill=1;
}/**if(refill)**/
else{
strcpy(array,"");
if(refill==0){
firstk_filer=firstk;icsr_filer=icsr;jcsr_filer=jcsr;cdflag=-1;  /* Esc */
}
else cdflag=0;                                          /* Shift + Esc */
refill_old=refill;
break;
}/**else(refill)**/
```

```
}/**while(1)**/

free(p[fn]);/*free(ptmp);*/
if(cdflag==0) chdir(home_global_GCD);
/*if(cdflag==0) SetCurrentDirectory(home_global_GCD);*/

end_:
setcsrcolor((csrcolor=CSRCOLOR));
filerflag=0;
refill=1;
}/** filer_tt **/


void sort(int count,int x)
{
}/** sort **/


void sort_dp(char flag,struct dirent *dp)
{
static int count=0,inimin=0xff,inimax=0;
int x;

if(flag==0){
dps[count]=dp;
count++;

if(dp->d_name[0]<inimin) inimin=dp->d_name[0];
if(dp->d_name[0]>inimax) inimax=dp->d_name[0];
}
else{
printf(" %s\n",dps[count-1]->d_name);
printf(" %c\n",inimin);
printf(" %c\n",inimax);
x=(inimin+inimax)/2;
/*sort(count,x);*/
printf(" %s\n",dps[0]->d_name);
printf(" %s\n",dps[count-1]->d_name);
}
}/** sort_dp **/


int make_list(unsigned char *home,unsigned char *fstr)
{
/*static count_nest=0;*/
int type,length,r,i,flag;
```

```
long kmax_list=0,kceil_list=ASIZE*2,dk_list;
char *str;
unsigned char *plist,*alloctmp,buf[ASIZE],buf_[ASIZE],buf_L[ASIZE];
struct dirent **dp;
struct stat sp;

#if GRP_or_EDT==0
strcpy(buf,home);
strcpy(buf_,home);
/*flag=strlen(buf_);

if(strlen(fstr)==0){
if(buf_[flag-1]!='/'){
strcat(buf_,"/.*");
}
else{
strcat(buf_,".*");
}
}*//**if(strlen(fstr))**/
/*else{
strcat(buf_,fstr);
}*//**else(strlen(fstr))**/
#else
getcwd(buf_,ASIZE);
if(FF_2/2) strcpy(fname_bg,buf_);
#endif

if((r=scandir(buf_,&dp,NULL,alphasort))==-1) return 2;

plist=(unsigned char *)malloc(sizeof(unsigned char)*(kceil_list+(1+1)));
if(plist==NULL) goto end_;

count_dir=count_file=0;

for(i=0;i<r;i++){
type=dp[i]->d_type;

stat(dp[i]->d_name,&sp);
str=ctime(&sp.st_mtime);
memmove(&str[5],&str[4],12);
memmove(&str[0],&str[20],4);str[17]='\0';
str[4]=' ';str[8]=' ';if(str[9]==' ') str[9]='0';


if(type!=DT_LNK){
dk_list=5+1+5+1+16+1+17+1+strlen(dp[i]->d_name)+1;
```

```
}
else{
length=readlink(dp[i]->d_name,buf_L,ASIZE);buf_L[length]='\0';
dk_list=5+1+5+1+16+1+17+1+strlen(dp[i]->d_name)+4+strlen(buf_L)+1;
}

if(kmax_list+dk_list>kceil_list){
kceil_list=(kmax_list+dk_list)*2-1;
alloctmp=(unsigned char *)realloc(plist,sizeof(unsigned char)*(kceil_list+(1+1)));
if(alloctmp!=NULL) plist=alloctmp;
else goto end;
}

if(type==DT_REG){ /* FILE */
sprintf(&plist[kmax_list],"-%04x %5d %16ld %s %s\n",sp.st_mode,sp.st_uid,sp.st_size,
                                        str,dp[i]->d_name);
count_file++;
}
else if(type==DT_DIR){ /* DIR */
sprintf(&plist[kmax_list],"d%04x %5d %16ld %s %s\n",sp.st_mode,sp.st_uid,sp.st_size,
                                        str,dp[i]->d_name);
count_dir++;
}
else if(type==DT_LNK) /* LNK */
sprintf(&plist[kmax_list],"l%04x %5d %16ld %s %s -> %s\n",sp.st_mode,sp.st_uid,
                                    sp.st_size,str,dp[i]->d_name,buf_L);
else /* others */
sprintf(&plist[kmax_list],"o%04x %5d %16ld %s %s\n",sp.st_mode,sp.st_uid,sp.st_size,
                                        str,dp[i]->d_name);

kmax_list+=dk_list;

#if GRP_or_EDT==0
if(type==DT_DIR){
if(to_sub==1){
if(strcmp(dp[i]->d_name,".")==0) goto next;
if(strcmp(dp[i]->d_name,"..")==0) goto next;

/*if(count_nest<10) count_nest++;
else goto next;*/

/*strcpy(array,dp[i]->d_name);*/
getcwd(buf_,ASIZE);
if(buf_[1]!='\0') strcat(buf_,"/");
strcat(buf_,dp[i]->d_name);
```

```
/*strcpy(array,"\ndir_in:");
strcat(array,buf_);
fprintf_(array);*/

/*arraycheck();*/
chdir(buf_);
make_list(buf_,fstr);
chdir(buf);

/*strcpy(array,"dir_out:");
strcat(array,buf_);
strcat(array,"\n");
fprintf_(array);

count_nest--;*/
}/**if(to_sub)**/
}
else if(type==DT_REG){
/*fprintf_(dp[i]->d_name);*/
flag=fgrep(dp[i]->d_name);
if(flag==0){
free(p[fn]);
fload_failed();
}
else if(flag==1){
fload_failed();
}
}
#endif

next:{}
}/*for(i)*/

#if GRP_or_EDT==1
fpf=fopen(home,"wb");
fwrite(&plist[0],1,kmax_list,fpf);
fclose(fpf);
#endif

free(plist);

return 0;

end:
free(plist);
```

```
end_:
for(i=0;i<r;i++) free(dp[i]);
free(dp);

return 1;
}/** make_list **/


int spacecheck(long member_,long member)
{
char checkflag;
int spacecount;

checkflag=1;
spacecount=0;
spaces=0;

while(1){
if(checkflag==1 && p[fn][member_]==' '){
                                                                        /* vine 4 */
if(MONTH==1 && member_>1 && p[fn][member_-2]==0xb7 && p[fn][member_-1]==0xee) ;
else{
checkflag=0;
spacecount++;
}
}

if(p[fn][member_]==' ') spaces++;

if(checkflag==0 && p[fn][member_]!=' '){
checkflag=1;
spaces=0;
}

if(member_==member) break;
member_++;
}/**while(1)**/

return spacecount;
}/** spacecheck **/


void autoindent(void)
{
char icsr_old,function_old;
char flag_;
```

```
long member;
long k,k_,dk_auto;

/*if(cut>0) {lumpflag=0;return;}*/

if(uflag==0)
member=topp[/*firstline+*/jcsr-1]+0;
else
member=topp[/*firstline+*/jcsr+1]+0;

if(p[fn][member]==0x09 || p[fn][member]==0x20){
k_=member;

while(1){
if(p[fn][member]!=0x09 && p[fn][member]!=0x20) break;
member++;
}

k=topp[/*firstline+*/jcsr]+0;
dk_auto=member-k_;
strncpy(buf_line,&p[fn][k_],dk_auto);

flag_=pdata_increase(k,&buf_line[0],dk_auto);

/*icsr_old=icsr;*/
function_old=function;function=2;
/*jcsr=*/while_puts_dline(/*first*/k,k+dk_auto);
function=function_old;

if(flag_==0) icsr=icsr_last;
/*else icsr=icsr_old;*/
if(flag_==0 && cut>0 && k<=k_from) k_from+=dk_auto;   /* <= */
}/**if(p[fn][member],p[fn][member])**/

lumpflag=0;
page_firstk(firstk);
}/** autoindent **/


int YKP_word(char operation)
{
char function_old;
long k,dk_word_old;

if(cut>0) return 1;
```

```
tailcheck();

k=top_icsr(/*firstline+*/jcsr,icsr);
if(k==kmax[fn]) return 1;

if(operation==0){                    /* 'Y' */
lumpflag=1;wordcheck(operation,k);if(lumpflag_global==0) lumpflag=0;
if(k_to-k_from>0 && k_to-k_from<=ASIZEM-1){
firstk_from=firstk;
icsr_from=icsr;jcsr_from=jcsr;
if(lumpflag_global==1) dk_word_old=dk_word;
dk_word=k_to-k_from;
if(lumpflag_global==0) memory(2);
text_to_file(5,0);
if(lumpflag_global==1) dk_word=dk_word_old;
else if(lumpflag_global==0) okflag_w=1;
deletion_dk();}

cut=0;
}/**if(operation)**/
else if(operation==1){               /* 'K' */
lumpflag=1;wordcheck(operation,k);lumpflag=0;
if(k_to-k_from>0 && k_to-k_from<=ASIZEM-1){
dk_word=k_to-k_from;
memory(2);
okflag_w=1;
page_firstk(firstk);beep(50);}

cut=0;
}/**else if(operation)**/
else{                                /* 'P' */
if(cut>0) return 1;

if(okflag_w){
lumpflag=1;
insertion_dk(2,k);
lumpflag=0;

if(MOVEcsr==1){
function_old=function;function=2;
jcsr=while_puts_dline(firstk,k+dk_word);
if(jcsr>ROW-1) {firstk=while_puts_firstk(firstk,jcsr-(ROW-1));jcsr=ROW-1;}
function=function_old;
icsr=icsr_last;}
page_firstk(firstk);
}
```

```
}/**else(operation)**/

return 0;
}/** YKP_word **/


void half_word(char flag)
{
lumpflag=1;

if(flag==0){
if(insertion(/*' '*/'!')==1) {lumpflag=0;goto end;}

lumpflag_global=1;
YKP_word(0);
lumpflag_global=0;
/*backspace();*/
if(csr_left()==0) deletion_onlymem();
}
else{
uflag=1;
if(insertion(/*' '*/'!')==1) {lumpflag=0;uflag=0;goto end;}
uflag=0;

if(csr_left()==0){
lumpflag_global=1;
YKP_word(0);
lumpflag_global=0;
/*deletion();*/
deletion_onlymem();
}
else deletion_onlymem();
}

lumpflag=0;
page_firstk(firstk);

end:{}
}/** half_word **/


void find_word(char flag)
{
char type_jp,type_jp_;
long k;
```

```
lumpflag=1;

tailcheck();

while(1){
if(flag==0) csr_right();
else csr_left();

k=top_icsr(/*firstline+*/jcsr,icsr);

if(flag==0) {if(k>=kmax[fn]) break;}
else {if(k==0) break;}

/* k<kmax[fn] && k>0 */
if(icsr==0 && p[fn][k-1]=='\n') break;
if(p[fn][k]=='\n') break;

                                        /* 0 : word */
                                        /* 1 : control code, symbol(1byte) */
                                        /* 2 : kana */
                                        /* 3 : kanji */
                                        /* 4 : katakana */
                                        /* 5 : hiragana */
                                        /* 6 : alphabet, figure */
                                        /* 7 : greek */
                                        /* 8 : tab, half space, full space */
                                        /* 9 : symbol(2bytes) */

type_jp=gettype_jp(k);

if(type_jp<=8 && type_jp!=1){

if(type_jp==3){                    /* kanji */
if(ishead(k-1)==0){
type_jp_=gettype_jp(k-1);
if(type_jp_==1 || type_jp_==8) break;}
else{
type_jp_=gettype_jp(k-2);
if(type_jp_==5 || type_jp_==9 || type_jp_==8) break;}
}
else if(type_jp==4){                    /* katakana */
if(ishead(k-1)==0){
type_jp_=gettype_jp(k-1);
if(type_jp_==1 || type_jp_==8) break;}
else{
type_jp_=gettype_jp(k-2);
```

```
if(type_jp_==5 || type_jp_==9 || type_jp_==8) break;}
}
else if(type_jp==5){                    /* hiragana */
if(ishead(k-1)==0){
type_jp_=gettype_jp(k-1);
if(type_jp_==1 || type_jp_==8) break;}
else{
type_jp_=gettype_jp(k-2);
if(/*type_jp_==5 || */type_jp_==9 || type_jp_==8) break;}
}
else if(type_jp==6){                    /* alphabet, figure */
if(ishead(k-1)==0){
type_jp_=gettype_jp(k-1);
if(type_jp_==1 || type_jp_==8) break;}
else{
type_jp_=gettype_jp(k-2);
if(type_jp_==5 || type_jp_==9 || type_jp_==8) break;}
}
else if(type_jp==7){                    /* greek */
if(ishead(k-1)==0){
type_jp_=gettype_jp(k-1);
if(type_jp_==1 || type_jp_==8) break;}
else{
type_jp_=gettype_jp(k-2);
if(type_jp_==5 || type_jp_==9 || type_jp_==8) break;}
}
else if(type_jp==8){                    /* tab, half space, full space */
if(tabspaces){
if(ishead(k-1)==0){
type_jp_=gettype_jp(k-1);
if(type_jp_==1/* || type_jp_==8*/) break;}
else{
type_jp_=gettype_jp(k-2);
if(type_jp_==5 || type_jp_==9/* || type_jp_==8*/) break;}
}
}
else{                                   /* type_jp==0, word */
if(ishead(k-1)==0){
type_jp_=gettype_jp(k-1);
if(type_jp_==1 || type_jp_==8) break;}
else{
type_jp_=gettype_jp(k-2);
if(type_jp_==5 || type_jp_==9 || type_jp_==8) break;}
}

}
```

```
}/**while(1)**/

lumpflag=0;
page_firstk(firstk);
}/** find_word **/


void find_0x1a(char flag)
{
int jcsr_ini;
long k;

if(flag==1) csr_row_home();
else csr_row_end();

return;



/*lumpflag=1;

tailcheck();

k=top_icsr(jcsr,icsr);
if(flag==1 && k>0 && ishead(k-1)==0 && p[fn][k-1]=='\n') csr_left();

while(1){
if(flag==0) csr_right();
else{
jcsr_ini=jcsr;
csr_left();
}

k=top_icsr(jcsr,icsr);

if(flag==0) {if(k>=kmax[fn]) break;}
else {if(k==0) break;}

if(p[fn][k]=='\n'){
if(flag==1){
csr_right();
while(1){
if(jcsr>jcsr_ini) scroll_down(1);
else break;
}
}
```

```
break;
}
}*//**while(1)**/

/*lumpflag=0;
page_firstk(firstk);*/
}/** find_0x1a **/


void half_line(char flag)
{
char type;

tailcheck();

firstk_from=firstk;
icsr_from=icsr;jcsr_from=jcsr;
if(flag==1) icsr_from=0;

if(flag==0){
k_from=top_icsr(/*firstline+*/jcsr,icsr);
k_to=top_icsr(/*firstline+*/jcsr,return_is(/*firstline+*/jcsr));
if(k_to!=kmax[fn]){
type=gettype_p(k_to);
if(type<=2) {if(p[fn][k_to]!='\n') k_to++;}
else if(type==3) k_to+=2;
else ;}
}
else{
k_from=topp[/*firstline+*/jcsr]+0;
k_to=top_icsr(/*firstline+*/jcsr,icsr);
}

if(k_to-k_from!=0){
/*swap_BL(0);*/
dk_old=dk;dk=k_to-k_from;
/*if(memory(0)==0){*/
text_to_file(4,0);
dk=dk_old;
/*dline=dk;
paste=2;okflag_BL=1;
cut=0;*/
deletion_dk();/*}*/}
}/** half_line **/
```

```
int wordcheck_unvisible(char operation,long k)
{
char type;
int jcsr_ini;
int icsr_old,jcsr_old;
long firstk_old;
long member,dmember;

member=k;

if(operation==1){
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;
}

type=gettype_p(member);
if(type<=2) dmember=1;
else dmember=2;

while(1){
member+=dmember;                    /* 1byte or 2bytes */

type=gettype_p(member);
if(type<=2){
if(
(type!=1 &&                         /* (!tab && !half space) || kmax */
 (/*type!=0 || */p[fn][member]!=0x20)) || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
dmember=1;
}/**if(type)**/
else if(type==3){
if(
                                                    /* !full space || kmax */
/*type!=3 || */p[fn][member]!=/*0xa1*/SPC1 || p[fn][member+1]!=/*0xa1*/SPC2
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
dmember=2;
}/**else if(type)**/
else{
}/**else(type)**/
}/**while(1)**/

jcsr_ini=jcsr;
```

```
while(1){
member=top_icsr(/*firstline+*/jcsr,icsr);
type=gettype_p(member);
if(type<=2){
if(
(type!=1 &&                          /* (!tab && !half space) */
 (/*type!=0 || */p[fn][member]!=0x20))
)
{k_from=member+1;csr_right();break;}  /* k_left */
else {if(member==0) {k_from=member;break;}}
}/**if(type)**/
else if(type==3){
if(
                                                                  /* !full space */
/*type!=3 || */p[fn][member]!=/*0xa1*/SPC1 || p[fn][member+1]!=/*0xa1*/SPC2
)
{k_from=member+2;csr_right();break;}  /* k_left */
else {if(member==0) {k_from=member;break;}}
}/**else if(type)**/
else{
}/**else(type)**/

jcsr_ini=jcsr;
csr_left();
}/**while(1)**/

if(operation==0){
while(1){
if(jcsr>jcsr_ini) scroll_down(1);
else break;
}
}
else{
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
}

return 0;
}/** wordcheck_unvisible **/


int wordcheck_2bytes(char flag,char operation,long k)
{
char type;
int jcsr_ini;
```

```
int icsr_old,jcsr_old;
long firstk_old;
long member;

member=k;

if(operation==1){
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;
}

while(1){
member+=2;                          /* 2bytes */

type=gettype_p(member);
if(flag==0){
if(
                                                                    /* !kanji */
type!=3 || (p[fn][member]<0xb0 && (p[fn][member]!=0xa1 || p[fn][member+1]!=0xa8))
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
}
else if(flag==1){
if(
type!=3 || ((p[fn][member]!=0xa5 || p[fn][member+1]>0xf6)  /* !katakana */
            && (p[fn][member]!=0xa1 || p[fn][member+1]!=0xbc))
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
}
else if(flag==2){
if(
type!=3 || p[fn][member]!=0xa4 || p[fn][member+1]<0xa1  /* !hiragana */
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
}
else if(flag==3){
if(
type!=3 || p[fn][member]!=0xa3 || p[fn][member+1]>0xfa  /* !alphabet, !figure */
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
}
else{
```

```
if(
type!=3 || p[fn][member]!=0xa6                          /* !greek */
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
}
}/**while(1)**/

jcsr_ini=jcsr;

while(1){
member=top_icsr(/*firstline+*/jcsr,icsr);
type=gettype_p(member);
if(type<=2){
k_from=member+1;csr_right();break;  /* k_left */
}/**if(type)**/
else if(type==3){
if(flag==0){
if(
/*type!=3 || */
(p[fn][member]<0xb0 && (p[fn][member]!=0xa1 || p[fn][member+1]!=0xa8)) /* !kanji */
)
{k_from=member+2;csr_right();break;}  /* k_left */
else {if(member==0) {k_from=member;break;}}
}
else if(flag==1){
if(
/*type!=3 || */((p[fn][member]!=0xa5 || p[fn][member+1]>0xf6)  /* !katakana */
            && (p[fn][member]!=0xa1 || p[fn][member+1]!=0xbc))
)
{k_from=member+2;csr_right();break;}  /* k_left */
else {if(member==0) {k_from=member;break;}}
}
else if(flag==2){
if(
/*type!=3 || */p[fn][member]!=0xa4 || p[fn][member+1]<0xa1  /* !hiragana */
)
{k_from=member+2;csr_right();break;}  /* k_left */
else {if(member==0) {k_from=member;break;}}
}
else if(flag==3){
if(
/*type!=3 || */p[fn][member]!=0xa3 || p[fn][member+1]>0xfa  /* !alphabet, !figure */
)
{k_from=member+2;csr_right();break;}  /* k_left */
else {if(member==0) {k_from=member;break;}}
```

```
}
else{
if(
/*type!=3 || */p[fn][member]!=0xa6                              /* !greek */
)
{k_from=member+2;csr_right();break;}  /* k_left */
else {if(member==0) {k_from=member;break;}}
}
}/**else if(type)**/
else{
}/**else(type)**/

jcsr_ini=jcsr;
csr_left();
}/**while(1)**/

if(operation==0){
while(1){
if(jcsr>jcsr_ini) scroll_down(1);
else break;
}
}
else{
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
}

return 0;
}/** wordcheck_2bytes **/


int wordcheck_kana(char operation,long k)
{
return 0;
}/** wordcheck_kana **/


int wordcheck(char operation,long k)
{
char type;
int jcsr_ini;
int icsr_old,jcsr_old;
long firstk_old;
long member;

if(k==kmax[fn]) {k_to=k_from=k;return 1;}
```

```
member=k;
type=gettype_p(member);

if(type!=0 ||
   (p[fn][member]<0x30 && p[fn][member]!=0x24) ||
   (p[fn][member]>0x39 && p[fn][member]<0x41) ||
   (p[fn][member]>0x5A && p[fn][member]<0x5F) ||
    p[fn][member]>0x7A || p[fn][member]==0x60
)
{                                   /* control code, symbol(1byte), 2bytes */
if(
                                                                        /* kanji */
type==3 && (p[fn][member]>=0xb0 || (p[fn][member]==0xa1 && p[fn][member+1]==0xa8))
)
wordcheck_2bytes(0,operation,k);
else if(
type==3 && ((p[fn][member]==0xa5 && p[fn][member+1]<=0xf6)  /* katakana */
            || (p[fn][member]==0xa1 && p[fn][member+1]==0xbc))
)
wordcheck_2bytes(1,operation,k);
else if(
type==3 && p[fn][member]==0xa4 && p[fn][member+1]>=0xa1  /* hiragana */
)
wordcheck_2bytes(2,operation,k);
else if(
type==3 && p[fn][member]==0xa3 && p[fn][member+1]<=0xfa  /* alphabet, figure */
)
wordcheck_2bytes(3,operation,k);
else if(
type==3 && p[fn][member]==0xa6      /* greek */
)
wordcheck_2bytes(4,operation,k);
else if(
tabspaces==1 && ((type==1) ||       /* tab, half space, full space */
(type==0 && p[fn][member]==0x20) ||
(type==3 && p[fn][member]==/*0xa1*/SPC1 && p[fn][member+1]==/*0xa1*/SPC2))
)
wordcheck_unvisible(operation,k);
else{                               /* control code, symbol(1byte, 2bytes) */
/*k_to=k_from=member;
if(operation==0) {if(lumpflag_global==0) lumpflag=0;deletion();}*/

if(type<=2) {k_from=member;k_to=member+1;}
else {k_from=member;k_to=member+2;}
}
```

```
return 1;
}/**if(type,p[fn][member])**/

if(operation==1){
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;
}

while(1){
member++;                          /* 1byte */

type=gettype_p(member);
if(type!=0 ||
   (p[fn][member]<0x30 && p[fn][member]!=0x24) ||
   (p[fn][member]>0x39 && p[fn][member]<0x41) ||
   (p[fn][member]>0x5A && p[fn][member]<0x5F) ||
    p[fn][member]>0x7A || p[fn][member]==0x60
                      || member==kmax[fn]
)
{k_to=member;break;}               /* k_right */
}/**while(1)**/

jcsr_ini=jcsr;

while(1){
member=top_icsr(/*firstline+*/jcsr,icsr);
type=gettype_p(member);
if(type<=2){
if(type!=0 ||
   (p[fn][member]<0x30 && p[fn][member]!=0x24) ||
   (p[fn][member]>0x39 && p[fn][member]<0x41) ||
   (p[fn][member]>0x5A && p[fn][member]<0x5F) ||
    p[fn][member]>0x7A || p[fn][member]==0x60
)
{k_from=member+1;csr_right();break;}  /* k_left */
else {if(member==0) {k_from=member;break;}}
}/**if(type)**/
else if(type==3){
k_from=member+2;csr_right();break;  /* k_left */
}/**else if(type)**/
else{
}/**else(type)**/

jcsr_ini=jcsr;
csr_left();
```

```
}/**while(1)**/

if(operation==0){
while(1){
if(jcsr>jcsr_ini) scroll_down(1);
else break;
}
}
else{
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
}

return 0;                              /* word */
}/** wordcheck **/


unsigned char subroop(void)
{
charflag=1;

while(1){
kbhit_();
if(charflag==0) return charcode;
}
}/** subroop **/


void mainroop(void)
{
while(1){
kbhit_();
if(refill!=1) break;

while(1){
if(nest_free_flag==1){
nest_free_flag=0;

  if(/*nest<NEST*/1){
  nest++;
  function=1;if(GKS_(ControlMask)<0 && fn!=FMAX-1) u_s_flag=1;else u_s_flag=0;
  if(GKS_(ShiftMask)<0) dlgproc_REF(1);else dlgproc_REF(0);
  function=0;if(nestflag) {nestflag=0;monitorline(1);BitBltflag=2;}
  nest=0;}
  /*else nest_free();*/
```

```
if(fn!=FMAX-1 && ftp>0) write_3vals(ftp-1);

if(BitBltflag==0)      {BitBlt_full();csr();}
else if(BitBltflag==1) {monitorline(1);csr();}
else{}
}
else break;
}/**while(1)**/
}
}/** mainroop **/


void csr_to_1(void)
{
int dy=0;
long k;

XSetFunction(d,gcdisplay,GXxor);
bitbltflag=1;

if(dialogflag==0){
if(menuflag==1)
bitblt(-3,0*UDX,0*UDY,sizeoffname*UDX+1,UDY,
         (4+DI_m+DI)*UDX-1,(jcsr+DJ)*UDY+dy);  /* mnuproc_MULTIFILE */
else if(menuflag==2)
bitblt(-3,0*UDX,0*UDY,12*UDX+1,UDY,
         (3+DI_m+DI)*UDX-1,(jcsr+DJ)*UDY+dy);  /* mnuproc_REP */
else if(filerflag==1)
bitblt(-3,0*UDX,0*UDY,FCSRSIZE*UDX,UDY,
         (0+DI)*UDX,(jcsr+DJ)*UDY+dy);  /* filer */
else{
if(cut==2 && jcsr_f==jcsr && icsr_f==icsr) goto skip;

if(icsr<=return_is(/*firstline+*/jcsr)){
k=top_icsr(/*firstline+*/jcsr,icsr);
if(flag_2nd==0){
if(gettype_p(k)!=3)
bitblt(-3,0*UDX,0*UDY,UDX,CSRDY,                    /* text(single byte) */
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);
else
bitblt(-3,0*UDX,0*UDY,UDX*2,CSRDY,              /* text(double byte,1st) */
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);
}
else
bitblt(-3,0*UDX,0*UDY,UDX*2,CSRDY,              /* text(double byte,2nd) */
         (icsr-1+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);
```

```
}/**if(icsr)**/
else{
bitblt(-3,0*UDX,0*UDY,UDX,CSRDY,                    /* text(over return) */
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);
}/**else(icsr)**/

skip:{}
}
}/**if(dialogflag)**/
else{
k=firstk_dialog+icsr;
if(ishead_dialog(k)==0){
if(gettype_dialog(k)!=3)
bitblt(-3,0*UDX,0*UDY,UDX,CSRDY,                  /* dialog(single byte) */
         (icsr+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);
else
bitblt(-3,0*UDX,0*UDY,UDX*2,CSRDY,           /* dialog(double byte,1st) */
         (icsr+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);
}
else
bitblt(-3,0*UDX,0*UDY,UDX*2,CSRDY,           /* dialog(double byte,2nd) */
         (icsr-1+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);
}/**else(dialogflag)**/

XSetFunction(d,gcdisplay,GXcopy);
bitbltflag=0;
}/** csr_to_1 **/


void csr(void)
{
int dy=0;
long k;

/*XSetFunction(d,gcdisplay,GXxor);*/
/*bitbltflag=1;*/

if(dialogflag==0 && menuflag==0 && filerflag==0){
if(cut>0) indicator(1);
else {if(indicationflag) {indicationflag=0;indicator(0);}}
}
else BitBlt_indicator();

csr_to_1();

if(dialogflag==0){
```

```
if(menuflag==1)
bitblt(3,0*UDX,0*UDY,sizeoffname*UDX+1,UDY,
         (4+DI_m+DI)*UDX-1,(jcsr+DJ)*UDY+dy);  /* mnuproc_MULTIFILE */
else if(menuflag==2)
bitblt(3,0*UDX,0*UDY,12*UDX+1,UDY,
         (3+DI_m+DI)*UDX-1,(jcsr+DJ)*UDY+dy);  /* mnuproc_REP */
else if(filerflag==1)
bitblt(3,0*UDX,0*UDY,FCSRSIZE*UDX,UDY,
         (0+DI)*UDX,(jcsr+DJ)*UDY+dy);  /* filer */
else{
if(cut==2 && jcsr_f==jcsr && icsr_f==icsr) goto skip;

if(icsr<=return_is(/*firstline+*/jcsr)){
k=top_icsr(/*firstline+*/jcsr,icsr);
if(flag_2nd==0){
if(gettype_p(k)!=3)
bitblt(3,0*UDX,0*UDY,UDX,CSRDY,                    /* text(single byte) */
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);
else
bitblt(3,0*UDX,0*UDY,UDX*2,CSRDY,              /* text(double byte,1st) */
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);
}
else
bitblt(3,0*UDX,0*UDY,UDX*2,CSRDY,              /* text(double byte,2nd) */
         (icsr-1+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);
}/**if(icsr)**/
else{
bitblt(3,0*UDX,0*UDY,UDX,CSRDY,                    /* text(over return) */
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);
}/**else(icsr)**/

skip:{}
}
}/**if(dialogflag)**/
else{
k=firstk_dialog+icsr;
if(ishead_dialog(k)==0){
if(gettype_dialog(k)!=3)
bitblt(3,0*UDX,0*UDY,UDX,CSRDY,                  /* dialog(single byte) */
         (icsr+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);
else
bitblt(3,0*UDX,0*UDY,UDX*2,CSRDY,            /* dialog(double byte,1st) */
         (icsr+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);
}
else
bitblt(3,0*UDX,0*UDY,UDX*2,CSRDY,            /* dialog(double byte,2nd) */
```

```
         (icsr-1+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);
}/**else(dialogflag)**/

/*XSetFunction(d,gcdisplay,GXcopy);*/
/*bitbltflag=0;*/

csr_to_1();

BitBltflag=0;
BitBltflag_=0;
}/** csr **/


void centering_theline(void)
{
long ddline;

within_linemax();

ddline=get_firstk(topp[jcsr],ROW/2);      /* ROW/2 <-> jcsr_ini */
if(ddline<0) jcsr=ddline+ROW/2;
else jcsr=ROW/2;
page_firstk(firstk);
}/** centering_theline **/


void centering_csr(void)
{
jcsr=ROW/2;

within_linemax();
}/** centering_csr **/


long max(long a,long b)
{
return (a>b)?a:b;
}/** max **/


long min(long a,long b)
{
return (a<b)?a:b;
}/** min **/
```

```
int reference_lump(char reffunc)
{
char first,string,left,right;
int jcsr_ini;
long member,member_,member_t,member_t_;

if(shorten()==1) return 1;
if(reffunc==0) reffunc=reffunc_global;
reffunc_REP=reffunc;
member_t=strlen(ref_t)-1;
member_t_=strlen(rep_t_)-1;

first=2;

tailcheck();
if(cut==0) jcsr_ini=jcsr; else jcsr_ini=jcsr_from;

/*if(cut==2 && k_to-k_from>0){
if(icsr>0)
member=top_icsr(jcsr,icsr-1);
else
member=top_icsr(jcsr,icsr);
}
else member=top_icsr(jcsr,icsr);*/
if(cut) member=k_to;  /* new member= */
else    member=top_icsr(jcsr,icsr);

if(allflag==1) member=0;
member_last=member;
allflag=0;lumpflag=0;

while(1){

if(function==1){
while(11){
direction=subroop();
if(direction<=2) break;
}/**while(11)**/
}/**if(function==1)**/

if(direction==2){                   /* ESCAPE */
repout:

charflag=0;charcode=2;
beep(50);
break;
```

```
}/**if(direction==2)**/

if(function==1){                    /* ? */
tailcheck();                        /* moved */
jcsr_ini=jcsr;
member=top_icsr(/*line*/jcsr,icsr);
}

while(2){
if(first==2){
first=1;
}
else{
if(first==1) first=0;
}

if(direction==1){                   /* NEXT */

if(member==kmax[fn]){                  /* floor */
if(function==2){
while_puts_fload_(1,jcsr_ini);
rependflag=1;
page_firstk(firstk);
rependflag=0;
/*csr();*/

direction=2;
break;
}

/*BitBlt_full();csr();*/
beep(50);/*delay_(100);beep(50);*/

break;
}

member_=0;
while(11){
if(member+member_==kmax[fn]) break;

if(l_s_flag==1){
if(large_small(member,member_)==0) {}
else break;
if(member_==0 && ishead(member)!=0) break;
}/**if(l_s_flag)**/
else{
```

```
if(p[fn][member+member_]!=ref_t[member_]) break;
}/**else(l_s_flag)**/


member_++;
if(member_==member_t+1) break;
}/**while(11)**/

if(member_==member_t+1)
string=1;
else
string=0;

if(string==1 && ishead(member)==0 && ishead(member+member_t+1)==0){  /* if_1 */
if((reffunc==0 &&
    (member==0 ||
     (p[fn][member-1]<0x30 && p[fn][member-1]!=0x24) ||
     (p[fn][member-1]>0x39 && p[fn][member-1]<0x41) ||
     (p[fn][member-1]>0x5A && p[fn][member-1]<0x5F) ||
      p[fn][member-1]>0x7A || p[fn][member-1]==0x60 || ishead(member-1)!=0
    )
   ) || reffunc==1
  )
left=1;
else
left=0;

if(left){                           /* if_2 */
if((reffunc==0 &&
    (member+member_t+1==kmax[fn] ||
     (p[fn][member+member_t+1]<0x30 && p[fn][member+member_t+1]!=0x24) ||
     (p[fn][member+member_t+1]>0x39 && p[fn][member+member_t+1]<0x41) ||
     (p[fn][member+member_t+1]>0x5A && p[fn][member+member_t+1]<0x5F) ||
      p[fn][member+member_t+1]>0x7A || p[fn][member+member_t+1]==0x60
    )
   ) || reffunc==1
  )
right=1;
else
right=0;

if(right){                          /* if_3 */
/*icsr=member_top(member,line);
page_firstline(line-jcsr_ini);
if(line-jcsr_ini<0) jcsr=line;
else jcsr=jcsr_ini;
```

```
csr();*/

/*if(function==2){
while(11){
yorn=subroop();
if(yorn<=2) break;
}

if(yorn==2){
direction=2;
break;
}

if(yorn==0){*/
if(cut>0 && member<k_from){
if(member+member_t<k_from) k_from+=member_t_-member_t;
else {member=k_from;goto floor;}
}

repcount++;
if(replacement_lump(member,member_t,member_t_)) {flag_global=1;goto repout;}
member=member_last;
/*}
}*//**if(function)**/

/*break;*/                          /* notice ! */
}/**if_3**/
}/**if_2**/
}/**if_1**/

member++;

floor:

if(cut>0 && member==k_from) member=kmax[fn];
}/**if(direction)*********************************************************/
else{                               /* PRIOR */
member_=0;
while(11){
if(member+member_==kmax[fn]) break;

if(l_s_flag==1){
if(large_small(member,member_)==0) {}
else break;
if(member_==0 && ishead(member)!=0) break;
}/**if(l_s_flag)**/
```

```
else{
if(p[fn][member+member_]!=ref_t[member_]) break;
}/**else(l_s_flag)**/


member_++;
if(member_==member_t+1) break;
}/**while(11)**/

if(member_==member_t+1)
string=1;
else
string=0;

if(string==1 && ishead(member)==0 && ishead(member+member_t+1)==0){  /* if_1 */
if((reffunc==0 &&
    (member==0 ||
     (p[fn][member-1]<0x30 && p[fn][member-1]!=0x24) ||
     (p[fn][member-1]>0x39 && p[fn][member-1]<0x41) ||
     (p[fn][member-1]>0x5A && p[fn][member-1]<0x5F) ||
      p[fn][member-1]>0x7A || p[fn][member-1]==0x60 || ishead(member-1)!=0
    )
   ) || reffunc==1
  )
left=1;
else
left=0;

if(left){                           /* if_2 */
if((reffunc==0 &&
    (member+member_t+1==kmax[fn] ||
     (p[fn][member+member_t+1]<0x30 && p[fn][member+member_t+1]!=0x24) ||
     (p[fn][member+member_t+1]>0x39 && p[fn][member+member_t+1]<0x41) ||
     (p[fn][member+member_t+1]>0x5A && p[fn][member+member_t+1]<0x5F) ||
      p[fn][member+member_t+1]>0x7A || p[fn][member+member_t+1]==0x60
    )
   ) || reffunc==1
  )
right=1;
else
right=0;

if(right){                          /* if_3 */
/*icsr=member_top(member,line);
page_firstline(line-jcsr_ini);
if(line-jcsr_ini<0) jcsr=line;
```

```
else jcsr=jcsr_ini;
csr();*/

/*if(function==2){
while(11){
yorn=subroop();
if(yorn<=2) break;
}

if(yorn==2){
direction=2;
break;
}

if(yorn==0){*/
repcount++;
if(replacement_lump(member,member_t,member_t_)) {flag_global=1;goto repout;}
member=member_last;
/*}
}*//**if(function)**/

/*break;*/                          /* notice ! */
}/**if_3**/
}/**if_2**/
}/**if_1**/

ceil:

if(cut>0 && member==k_from && first==0) member=0;

if(member==0){                      /* ceil */
if(function==2){
while_puts_fload_(1,jcsr_ini);
rependflag=1;
page_firstk(firstk);
rependflag=0;
/*csr();*/

direction=2;
break;
}

/*BitBlt_full();csr();*/
beep(50);/*delay_(100);beep(50);*/

break;
```

```
}

member--;
}/**else(direction)**/
}/**while(2)**/
}/**while(1)**/

return 0;
}/** reference_lump **/


int replacement_lump(long member,long member_t,long member_t_)
{
char flag_;

if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;

if(member_t_<0){
if(direction==1 && member>0) member_last=member-1;  /* moved and modified */
else member_last=member;
}
k_from_rep=member;k_to_rep=member+member_t+1;
deletion_dk_lump();

if(member_t_<0){
return 0;
}

if(direction==1) member_last=member+member_t_+1-1;
else member_last=member;
flag_=insertion_dk_lump(member,member_t_);

if(flag_==0){
return 0;
}
else{
return 1;
}
}/** replacement_lump **/


int large_small(long member,long member_)
{
char flag,flag_;

flag_=0;
```

```
if(member_==0) flag=0;
else flag=ishead_(member,member_);

if(flag==0 &&
   ((p[fn][member+member_]>=0x41 && p[fn][member+member_]<=0x5A) ||
    (p[fn][member+member_]>=0x61 && p[fn][member+member_]<=0x7A))){
if(p[fn][member+member_]>=0x41 && p[fn][member+member_]<=0x5A){
if(p[fn][member+member_]==ref_t[member_] ||
   p[fn][member+member_]==ref_t[member_]-0x20) {}
else /*break*/flag_=1;
}
else{
if(p[fn][member+member_]==ref_t[member_] ||
   p[fn][member+member_]==ref_t[member_]+0x20) {}
else /*break*/flag_=1;
}
}
else{
if(p[fn][member+member_]!=ref_t[member_]) /*break*/flag_=1;
}

if(flag_==0) return 0;
else return 1;
}/** large_small **/


int reference(char reffunc)
{
char first,string,left,right;
int jcsr_ini;
long member,member_,member_t,member_t_,val;

if(shorten()==1) return 1;
if(reffunc==0) reffunc=reffunc_global;
if(function==1) reffunc_REF=reffunc;
else reffunc_REP=reffunc;
member_t=strlen(ref_t)-1;
member_t_=strlen(rep_t_)-1;

#if GRP_or_EDT==1
if(function==1) extraline(1);
#endif

first=3;
```

```
#if GRP_or_EDT==0
jcsr_ini=0;
if(cut) member=k_to;  /* new member= */
else    member=0;
#else
tailcheck();                        /* moved */
jcsr_ini=jcsr;
if(cut) member=k_to;  /* new member= */
else    member=top_icsr(jcsr,icsr);
#endif
if(allflag==1) member=0;
member_last=member;

#if GRP_or_EDT==1
if(function==1) monitorline(1);     /* icsr, reffunc */
#endif

while(1){

if(function==1){
if(fn!=FMAX-1) write_3vals(ftp-1);

direction_old=direction;
#if GRP_or_EDT==0
direction=1;
#else
while(11){
direction=subroop();
if(direction<=2) break;
}/**while(11)**/
#endif

if(direction!=direction_old) first=3;
if(filer_execute_phantom) {filer_execute_phantom=0;first=2;}
}/**if(function==1)**/

if(direction==2){                   /* ESCAPE */
repout:

charflag=0;charcode=2;

nestflag=0;
if(nest>0 && refill==1) {beep(50);nestflag=1;}
break;
}/**if(direction==2)**/
```

```
#if GRP_or_EDT==1
if(function==1){
tailcheck();                          /* moved */
jcsr_ini=jcsr;
member=top_icsr(jcsr,icsr);
}
#endif

if(first==3){
first=2;
}/**if(first)**/
else{
if(direction==1){                     /* NEXT */
if(member<kmax[fn]) member++;
if(cut>0 && member==k_from) goto floor;
}

if(direction==0){                     /* PRIOR */
if(cut>0 && member==k_from && first==0) goto ceil;
if(member>0) member--;
}
}/**else(first)**/

while(2){
if(first==2){
first=1;
}
else{
if(first==1) first=0;
}

if(direction==1){                     /* NEXT */

if(member==kmax[fn]){                    /* floor */
if(function==2){
while_puts_fload_(1,jcsr_ini);
rependflag=1;
page_firstk(firstk);
rependflag=0;
/*csr();*/

direction=2;
break;
}

#if GRP_or_EDT==0
```

```
return 1;
#else
beep(50);break;
#endif
}

member_=0;
while(11){
if(member+member_==kmax[fn]) break;

if(l_s_flag==1){
if(large_small(member,member_)==0) {}
else break;
if(member_==0 && ishead(member)!=0) break;
}/**if(l_s_flag)**/
else{
if(p[fn][member+member_]!=ref_t[member_]) break;
}/**else(l_s_flag)**/


member_++;
if(member_==member_t+1) break;
}/**while(11)**/

if(member_==member_t+1)
string=1;
else
string=0;

if(string==1 && ishead(member)==0 && ishead(member+member_t+1)==0){  /* if_1 */
if((reffunc==0 &&
    (member==0 ||
     (p[fn][member-1]<0x30 && p[fn][member-1]!=0x24) ||
     (p[fn][member-1]>0x39 && p[fn][member-1]<0x41) ||
     (p[fn][member-1]>0x5A && p[fn][member-1]<0x5F) ||
      p[fn][member-1]>0x7A || p[fn][member-1]==0x60 || ishead(member-1)!=0
    )
   ) || reffunc==1
  )
left=1;
else
left=0;

if(left){                           /* if_2 */
if((reffunc==0 &&
    (member+member_t+1==kmax[fn] ||
```

```
     (p[fn][member+member_t+1]<0x30 && p[fn][member+member_t+1]!=0x24) ||
     (p[fn][member+member_t+1]>0x39 && p[fn][member+member_t+1]<0x41) ||
     (p[fn][member+member_t+1]>0x5A && p[fn][member+member_t+1]<0x5F) ||
      p[fn][member+member_t+1]>0x7A || p[fn][member+member_t+1]==0x60
    )
   ) || reffunc==1
  )
right=1;
else
right=0;

if(right){                          /* if_3 */
member_global=member;
if(function==1) while_puts_theline(jcsr_ini);          /* -> firstk, jcsr, icsr */
else while_puts_fload_(0,jcsr_ini);
#if GRP_or_EDT==0
while_puts_show_(0,firstk);
strncpy(GRP_line,&p[fn][firstk],k_g-firstk);GRP_line[k_g-firstk]='\0';
val=while_puts_linenumber(0,firstk)+1;
if(GRP_line[k_g-firstk-1]=='\n'){
printf("%s %ld:",fname,val);
printf("%s",GRP_line);
}
else{
printf("%s %ld:",fname,val);
printf("%s\n",GRP_line);
}
#else
page_firstk(firstk);
csr();
#endif

if(function==2){
if(fn!=FMAX-1) write_3vals(ftp-1);
puts_mline(2,"? (Y/N)");
while(11){
yorn=subroop();
if(yorn<=2) break;
}/**while(11)**/

if(yorn==2){
while_puts_fload_(1,jcsr_ini);
rependflag=1;
page_firstk(firstk);
rependflag=0;
/*csr();*/
```

```
direction=2;
break;
}

if(yorn==0){
if(cut>0 && member<k_from){
if(member+member_t<k_from) k_from+=member_t_-member_t;
else {member=k_from;goto floor;}
}

repcount++;
if(replacement(member,member_t,member_t_)) {flag_global=1;goto repout;}
member=member_last;
}
}/**if(function)**/

break;
}/**if_3**/
}/**if_2**/
}/**if_1**/

member++;

floor:

if(cut>0 && member==k_from) member=kmax[fn];
}/**if(direction)************************************************************/
else{                                /* PRIOR */
member_=0;
while(11){
if(member+member_==kmax[fn]) break;

if(l_s_flag==1){
if(large_small(member,member_)==0) {}
else break;
if(member_==0 && ishead(member)!=0) break;
}/**if(l_s_flag)**/
else{
if(p[fn][member+member_]!=ref_t[member_]) break;
}/**else(l_s_flag)**/


member_++;
if(member_==member_t+1) break;
}/**while(11)**/
```

```
if(member_==member_t+1)
string=1;
else
string=0;

if(string==1 && ishead(member)==0 && ishead(member+member_t+1)==0){  /* if_1 */
if((reffunc==0 &&
    (member==0 ||
     (p[fn][member-1]<0x30 && p[fn][member-1]!=0x24) ||
     (p[fn][member-1]>0x39 && p[fn][member-1]<0x41) ||
     (p[fn][member-1]>0x5A && p[fn][member-1]<0x5F) ||
      p[fn][member-1]>0x7A || p[fn][member-1]==0x60 || ishead(member-1)!=0
    )
   ) || reffunc==1
  )
left=1;
else
left=0;

if(left){                           /* if_2 */
if((reffunc==0 &&
    (member+member_t+1==kmax[fn] ||
     (p[fn][member+member_t+1]<0x30 && p[fn][member+member_t+1]!=0x24) ||
     (p[fn][member+member_t+1]>0x39 && p[fn][member+member_t+1]<0x41) ||
     (p[fn][member+member_t+1]>0x5A && p[fn][member+member_t+1]<0x5F) ||
      p[fn][member+member_t+1]>0x7A || p[fn][member+member_t+1]==0x60
    )
   ) || reffunc==1
  )
right=1;
else
right=0;

if(right){                          /* if_3 */
member_global=member;
if(function==1) while_puts_theline(jcsr_ini);          /* -> firstk, jcsr, icsr */
else while_puts_fload_(0,jcsr_ini);
page_firstk(firstk);
csr();

if(function==2){
if(fn!=FMAX-1) write_3vals(ftp-1);
puts_mline(2,"? (Y/N)");
while(11){
yorn=subroop();
```

```
if(yorn<=2) break;
}/**while(11)**/

if(yorn==2){
while_puts_fload_(1,jcsr_ini);
rependflag=1;
page_firstk(firstk);
rependflag=0;
/*csr();*/

direction=2;
break;
}

if(yorn==0){
repcount++;
if(replacement(member,member_t,member_t_)) {flag_global=1;goto repout;}
member=member_last;
}
}/**if(function)**/

break;
}/**if_3**/
}/**if_2**/
}/**if_1**/

ceil:

if(cut>0 && member==k_from && first==0) member=0;

if(member==0){                      /* ceil */
if(function==2){
while_puts_fload_(1,jcsr_ini);
rependflag=1;
page_firstk(firstk);
rependflag=0;
/*csr();*/

direction=2;
break;
}

/*BitBlt_full();csr();*/
beep(50);/*delay_(100);beep(50);*/

break;
```

```
}

member--;
}/**else(direction)**/
}/**while(2)**/
}/**while(1)**/

return 0;
}/** reference **/


int replacement(long member,long member_t,long member_t_)
{
char flag_;

if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;

if(member_t_<0){
if(direction==1 && member>0) member_last=member-1;  /* moved and modified */
else member_last=member;
}
k_from_rep=member;k_to_rep=member+member_t+1;
deletion_dk_lump();

if(member_t_<0){
return 0;
}

if(direction==1) member_last=member+member_t_+1-1;
else member_last=member;
flag_=insertion_dk_lump(member,member_t_);

if(flag_==0){
return 0;
}
else{
return 1;
}
}/** replacement **/


int shorten(void)
{
long member_,length;

if(function==2) strcpy(ref_t,rep_s);
```

```
else strcpy(ref_t,ref_s);
if((length=strlen(ref_t))==0) return 1;

member_=0;
reffunc_global=0;

while(1){
if((ref_t[member_]<0x30 && ref_t[member_]!=0x24) ||
   (ref_t[member_]>0x39 && ref_t[member_]<0x41) ||
   (ref_t[member_]>0x5A && ref_t[member_]<0x5F) ||
    ref_t[member_]>0x7A || ref_t[member_]==0x60){
reffunc_global=1;break;}

member_++;
if(member_==length) break;
}

return 0;
}/** shorten **/


int shorten_(void)
{
long length_;

strcpy(rep_t_,rep_s_);
if((length_=strlen(rep_t_))==0) return 1;

return 0;
}/** shorten_ **/


int arraycheck(void)
{
char type;
int i,length,length_,val;
unsigned char buf_b[ASIZE],buf_b2[ASIZE];
unsigned char *pspace,curdir[ASIZE],curdir_m[ASIZE],tmp[ASIZE],
              tmp_[ASIZE],fullpath[ASIZE];

if((length=strlen(array))==0) return 2;  /* <- kmax_dialog==0 */

i=0;
/*while(1){
type=gettype_ac(i);
if(type<=2){
```

```
if(array[i]=='\\') array[i]='/';
i++;
}
else{
i+=2;
}

if(i==length) break;
}*/
while(1){
if(array[i]=='\\') array[i]='/';
i++;

if(i==length) break;
}

i=0;
while(1){
if(i==length-1) break;

if(array[i]=='/' && array[i+1]=='/') {strcpy(&array[i],&array[i+1]);length--;}
else i++;
}

if(array[0]!='/' && array[0]!='~'){
getcwd(tmp,ASIZE);
strcat(tmp,"/");
strcat(tmp,array);
strcpy(array,tmp);
/*printf(" %s\n",array);*/
}
else if(array[0]=='~'){
strcpy(tmp,&array[1]);
strcpy(array,home_global);
strcat(array,tmp);
}

if((length=strlen(array))==0) return 2;  /* <- kmax_dialog==0 */

getcwd(curdir_m,ASIZE);
chdir(home_global_GCD);

/*getcwd(curdir,ASIZE);*/
if(chdir(array)==0) {val=3;goto end_;}  /* OK_directory */

strcpy(tmp,array);
```

```
if((pspace=(unsigned char *)strrchr(tmp,'/'))!=NULL){
pspace++;
if(*pspace!='\0'){
strcpy(tmp_,pspace);                /* tmp_ : file name */
*pspace='\0';                       /* -> tmp : path */
}

length_=strlen(tmp);
if(length_==length){                /* only path */
if(chdir(tmp)==0) {val=3;goto end_;}  /* OK_directory : repeated */
else {val=1;goto end;}                    /* wrong_directory */
}
else{
getcwd(curdir,ASIZE);
if(chdir(tmp)==0){
getcwd(fullpath,ASIZE);
strcat(fullpath,"/");
strcat(fullpath,tmp_);
strcpy(array,fullpath);
chdir(curdir);val=0;goto end;}            /* OK_directory/file */
else {val=1;goto end;}                      /* wrong_directory/file */
}
}/**if(pspace)**/
else{
getcwd(fullpath,ASIZE);
strcat(fullpath,"/");
strcat(fullpath,tmp);
strcpy(array,fullpath);
val=0;goto end;                           /* OK_file */
}/**else(pspace)**/


end:
chdir(curdir_m);
end_:

i=0;
while(1){
if(i==length-1) break;

if(array[i]=='/' && array[i+1]=='/') {strcpy(&array[i],&array[i+1]);length--;}
else i++;
}

return val;
```

```
}/** arraycheck **/


char fopen_(void)
{
struct stat buf;
FILE *fp_;

if(newopen==1){
newopen=0;
return 0;
}

if(access(fname,0)==0){             /* exists */
if(access(fname,4)==-1) return 1;   /* unreadable */
#if GRP_or_EDT==0
openmode="rb";
#else
if(access(fname,2)==-1)             /* unwritable */
openmode="rb";
else
openmode="r+b";

if(strcmp(openmode,"r+b")==0){
fp_=fopen(fname,openmode);
if(fp_==NULL) openmode="rb";
else fclose(fp_);
}
#endif

unlinkflag=0;



stat(fname,&buf);

#if GRP_or_EDT==0
if(buf.st_size==0 || buf.st_size>mfsize*1024*1024){
return 1;
}
#else
if(buf.st_size>mfsize*1024*1024){
beep(500);
return 1;
}
#endif
```

```
else{
}



if(NKF==0){
if((fp=fopen(fname,openmode))==NULL) return 1;
}
else{
}
}/**if(access(fname,0))**/
else{
#if GRP_or_EDT==0
return 1;
#else
openmode="w+b";
unlinkflag=1;
if((fp=fopen(fname,openmode))==NULL) return 1;
#endif
}/**else(access(fname,0))**/

return 0;
}/** fopen_ **/



void insertion_u(void)
{
char flag_,lumpflag_old;
unsigned char charcode;
long k;
long span;

if(delsp>0){
span=getspan_u();

if(span==1){
delsp--;
charcode=stack_del[delsp];

if(stack_bs[delsp]==0){
uflag=1;
insertion(charcode);
uflag=0;
}
else{
/*uflag=1;*/
```

```
insertion(charcode);
/*uflag=0;*/
}
}/**if(span)**/
else if(span==2){
delsp-=2;

tailcheck();

k=top_icsr(/*firstline+*/jcsr,icsr);
flag_=pdata_increase(k,&stack_del[delsp],2);

if(flag_==0 && cut>0 && k<=k_from) k_from+=2;   /* <= */

if(flag_==0){
if(stack_bs[delsp]==0){
page_firstk(firstk);
}
else{
while_puts_show_(0,firstk);
/*lumpflag_old=lumpflag;lumpflag=1;*/
csr_right();
/*lumpflag=lumpflag_old;*/
page_firstk(firstk);
}

if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;
}
else{
page_firstk(firstk);
}
}/**else if(span)**/
else{
}/**else(span)**/
}/**if(delsp)**/
}/** insertion_u **/


void to_stack_2b(unsigned char deleted_1st,unsigned char deleted_2nd)
{
char type,type_,lessen;

lessen=0;

if(delorbs==0){
if(delsp==ASIZEM){
```

```
type=gettype_u(0);
if(type<=2) lessen=1;
else if(type==3) lessen=2;
else ;
}
else if(delsp==ASIZE){
type=gettype_u(0);
type_=gettype_u(1);
if(type<=2) {if(type_<=2) lessen=2;else if(type_==3) lessen=3;else;}
else if(type==3) lessen=2;
else ;
}
else{}

if(lessen>0){
memcpy(&stack_del[0],&stack_del[lessen],delsp-lessen);
memcpy(&stack_bs[0],&stack_bs[lessen],delsp-lessen);
delsp=delsp-lessen;}

stack_del[delsp]=deleted_1st;
stack_del[delsp+1]=deleted_2nd;
stack_bs[delsp]=0;                  /* del */
stack_bs[delsp+1]=0;
delsp+=2;
}/**if(delorbs)**/
else{
if(delsp==ASIZEM){
type=gettype_u(0);
if(type<=2) lessen=1;
else if(type==3) lessen=2;
else ;
}
else if(delsp==ASIZE){
type=gettype_u(0);
type_=gettype_u(1);
if(type<=2) {if(type_<=2) lessen=2;else if(type_==3) lessen=3;else;}
else if(type==3) lessen=2;
else ;
}
else{}

if(lessen>0){
memcpy(&stack_del[0],&stack_del[lessen],delsp-lessen);
memcpy(&stack_bs[0],&stack_bs[lessen],delsp-lessen);
delsp=delsp-lessen;}
```

```
stack_del[delsp]=deleted_1st;
stack_del[delsp+1]=deleted_2nd;
stack_bs[delsp]=1;                    /* bs */
stack_bs[delsp+1]=1;
delsp+=2;
}/**else(delorbs)**/
}/** to_stack_2b **/


void to_stack(unsigned char deleted)
{
if(delorbs==0){
if(delsp>ASIZEM){
memcpy(&stack_del[0],&stack_del[1],ASIZEM);
memcpy(&stack_bs[0],&stack_bs[1],ASIZEM);
delsp=ASIZEM;}

stack_del[delsp]=deleted;
stack_bs[delsp]=0;                    /* del */
delsp++;
}/**if(delorbs)**/
else{
if(delsp>ASIZEM){
memcpy(&stack_del[0],&stack_del[1],ASIZEM);
memcpy(&stack_bs[0],&stack_bs[1],ASIZEM);
delsp=ASIZEM;}

stack_del[delsp]=deleted;
stack_bs[delsp]=1;                    /* bs */
delsp++;
}/**else(delorbs)**/
}/** to_stack **/


int text_to_file(char flag,char flag_append)
{
char restore_file;
int existence,writable,length;
long dk_auto;
unsigned char bak[ASIZE];

/*if(deletedflag==1) return 1;
if(refflag==1) return 1;
if(filerflag==1) return 1;*/
if(fn==FMAX-1) return 1;
```

```
if(flag==0){                          /* file_SA:S,A in BL */
restore_file=0;

existence=access(file_SA,0);
writable=access(file_SA,2);

if(existence==0 && writable==-1) {/*beep(500);*/return 1;}

if(existence==0){
strcpy(bak,file_SA);length=strlen(bak);
if(length<ASIZEM-1){
bak[length]='~';
bak[length+1]='\0';

CopyFile(file_SA,bak);
/*unlink(file_SA);*/
}
else if(length==ASIZEM-1){
strcpy(bak,home_global);
strcat(bak,"zzz.$$$");

CopyFile(file_SA,bak);
/*unlink(file_SA);*/
}
else{}                                /* impossible */
}

if(flag_append) openmode="ab";
else openmode="wb";

if((fpf=fopen(file_SA,openmode))==NULL) restore_file=1;
else{
if((int)fwrite(&p[fn][k_from],1,dk_file,fpf)<dk_file){
message(-ferror(fpf),0);
clearerr(fpf);
restore_file=2;
}
fclose(fpf);
}

if(restore_file){
if(existence==0){
if(length<=ASIZEM-1){
if(restore_file==2) CopyFile(bak,file_SA);
unlink(bak);
}
```

```
else{}                              /* impossible */
}/**if(existence)**/
else{
if(restore_file==2) unlink(file_SA);
}/**else(existence)**/

/*if(restore_file==1) */return 1;
/*else return 0;*/
}/**if(restore_file)**/

/*beep(200);*/
}/**if(flag)**/
else if(flag==1){
fpf=fopen(home_ref,"ab");           /* zzz.find */
fwrite(&p[fn][k_from],1,dk_file,fpf);
if(p[fn][k_from+dk_file-1]!='\n' || ishead(k_from+dk_file-1)!=0)
  fwrite(&two[4][0],1,1,fpf);
fclose(fpf);
}/**else if(flag)**/
else{
if(d_or_t==0) fpf=fopen(home_deleted,"ab");  /* zzz.deleted with ^L*/
else          fpf=fopen(home_tmp,"ab");     /* zzz.string */

if(flag==2){                        /* line */
if(d_or_t==0) fwrite(&two[0][0],1,2,fpf);
if(d_or_t==0) dk_auto=dk_line;else dk_auto=dk_file;
fwrite(&p[fn][k_from],1,dk_auto,fpf);
if(jcsr==jcsrmax) fwrite(&two[4][0],1,1,fpf);
}
else if(flag==3){                   /* 'B' */
if(d_or_t==0) fwrite(&two[1][0],1,2,fpf);
if(d_or_t==0) dk_auto=dk;else dk_auto=dk_file;
fwrite(&p[fn][k_from],1,dk_auto,fpf);
}
else if(flag==4){                   /* 'L' */
if(d_or_t==0) fwrite(&two[2][0],1,2,fpf);
if(d_or_t==0) dk_auto=dk;else dk_auto=dk_file;
fwrite(&p[fn][k_from],1,dk_auto,fpf);
fwrite(&two[4][0],1,1,fpf);
}
else{                               /* word */
fwrite(&two[3][0],1,2,fpf);
dk_auto=dk_word;/*else dk_auto=dk_file;*/
fwrite(&p[fn][k_from],1,dk_auto,fpf);
fwrite(&two[4][0],1,1,fpf);
}
```

```
fclose(fpf);
}/**else(flag)**/

return 0;
}/** text_to_file **/


int file_to_text(void)              /* 'I' */
{
char flag_,reallocflag;
long k;

flag_=0;
reallocflag=0;

if((fpi=fopen(ins,"rb"))==NULL) return 1;
fseek(fpi,0L,2);
if((dk_ins=ftell(fpi))<1) {fclose(fpi);return 0;}
fseek(fpi,0L,0);

tailcheck();

if(paste==0 || paste==1){
k=topp[/*firstline+*/jcsr]+0;
}
else{
k=top_icsr(/*firstline+*/jcsr,icsr);
}

kmax[fn]+=dk_ins;if(kmax[fn]>kceil[fn]) reallocflag=p_realloc();
if(reallocflag==0){
memmove(&p[fn][k+dk_ins],&p[fn][k],(kmax[fn]-dk_ins)-(k)+1);

if((int)fread(&p[fn][k],1,dk_ins,fpi)<dk_ins){
message(-ferror(fpi),0);
clearerr(fpi);

fclose(fpi);
memmove(&p[fn][k],&p[fn][k+dk_ins],(kmax[fn])-(k+dk_ins)+1);
kmax[fn]-=dk_ins;
return 1;
}
fclose(fpi);
}/**if(reallocflag)**/
else{
```

```
flag_=2;
kmax[fn]-=dk_ins;
}/**else(reallocflag)**/

page_firstk(firstk);

if(flag_==0){
if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;
}
else{
message(7,0);
}

return 0;
}/** file_to_text **/


int backspace(void)
{
/*if(cut>0) return 1;*/

if(csr_left()==0) deletion();

return 0;
}/** backspace **/


char p_realloc(void)
{
long kceil_old;
unsigned char *alloctmp;

kceil_old=kceil[fn];

kceil[fn]=(kmax[fn]+1)+COLUMN*ROW_L-1;
alloctmp=(unsigned char *)realloc(p[fn],sizeof(unsigned char)*(kceil[fn]+(1+1)));

if(alloctmp!=NULL) {p[fn]=alloctmp;return 0;}
else {kceil[fn]=kceil_old;return 1;}
}/** p_realloc **/


char ptmp_realloc(void)
{
long kceiltmp_old;
unsigned char *alloctmp;
```

```
kceiltmp_old=kceiltmp;

kceiltmp=dk+0-1;
alloctmp=(unsigned char *)realloc(ptmp,sizeof(unsigned char)*(kceiltmp+(1+1)));

if(alloctmp!=NULL) {ptmp=alloctmp;return 0;}
else {kceiltmp=kceiltmp_old;return 1;}
}/** ptmp_realloc **/


int memory(char flag)
{
char flag_,reallocflag;

flag_=0;
reallocflag=0;

if(flag==0){                          /* ptmp, dk('B','L') */
/*if(dk-1>kceiltmp) */reallocflag=ptmp_realloc();
if(reallocflag==0) memmove(&ptmp[0],&p[fn][k_from],dk);
else {dk=dk_old;flag_=1;}
}
else if(flag==1){                     /* ptmp_line, dk_line */
memmove(&ptmp_line[0],&p[fn][k_from],dk_line);
}
else{                                 /* ptmp_word, dk_word */
memmove(&ptmp_word[0],&p[fn][k_from],dk_word);
}

if(flag_==0){
return 0;
}
else{
message(7,0);
return 1;
}
}/** memory **/


void tailcheck(void)
{
int ris;

if(/*firstline+*/jcsr>jcsrmax){
jcsr=jcsrmax;
```

```
ris=return_is(/*firstline+*/jcsr);
icsr=ris;
}
else{
ris=return_is(/*firstline+*/jcsr);
if(icsr>ris) icsr=ris;
}

csr_tab(0);
}/** tailcheck **/


int insertion(unsigned char charcode)
{
char flag_,reallocflag,lumpflag_old;
long k,dk=1;

/*if(cut>0) return 1;*/

tailcheck();

flag_=0;
reallocflag=0;

k=top_icsr(/*firstline+*/jcsr,icsr);

kmax[fn]+=dk;if(kmax[fn]>kceil[fn]) reallocflag=p_realloc();
if(reallocflag==0){
memmove(&p[fn][k+dk],&p[fn][k],kmax[fn]-dk-k+1);
p[fn][k]=charcode;
/*memmove(&p[fn][k],&pdk[0],dk);*/
}/**if(reallocflag)**/
else{
flag_=2;
kmax[fn]-=dk;
}/**else(reallocflag)**/

if(flag_==0 && cut>0 && k<=k_from) k_from+=1;   /* <= */

/*if(jcsr==ROW-1 && (charcode=='\n' || icsr==COLUMN-1)){
if(flag_==0 && uflag==0) {while_puts_show_(0,firstk);}
else
page_firstk(firstk);
}
else
page_firstk(firstk);*/
```

```
if(flag_==0){
while_puts_show_(0,firstk);
lumpflag_old=lumpflag;lumpflag=1;
if(uflag==0) csr_right();
lumpflag=lumpflag_old;
page_firstk(firstk);

if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;
return 0;
}
else{
message(7,2);
page_firstk(firstk);
return 1;
}
}/** insertion **/


char pdata_increase(long k,unsigned char *pdk,long dk)
{
char flag_,reallocflag;

flag_=0;
reallocflag=0;

kmax[fn]+=dk;if(kmax[fn]>kceil[fn]) reallocflag=p_realloc();
if(reallocflag==0){
memmove(&p[fn][k+dk],&p[fn][k],kmax[fn]-dk-k+1);
memmove(&p[fn][k],&pdk[0],dk);
}/**if(reallocflag)**/
else{
flag_=2;
kmax[fn]-=dk;
}/**else(reallocflag)**/

if(flag_==0){
return 0;
}
else{
message(7,/*2*/1);
if(filerflag==1 && lumpflag==2) {bitblt(1,0,0,XRESO,YRESO,0,0);/*BitBltflag=1;*/}
return 1;
}
}/** pdata_increase **/
```

```
char insertion_dk_lump(long member,long member_t_)
{
char flag_,reallocflag;
long k,dk_lump;

flag_=0;
reallocflag=0;

k=member;

dk_lump=member_t_+1;
kmax[fn]+=dk_lump;if(kmax[fn]>kceil[fn]) reallocflag=p_realloc();
if(reallocflag==0){
memmove(&p[fn][k+dk_lump],&p[fn][k],kmax[fn]-dk_lump-k+1);
memmove(&p[fn][k],&rep_t_[0],dk_lump);
}
else{
flag_=1;
}

if(flag_==0){
return 0;
}
else{
return 1;
}
}/** insertion_dk_lump **/


void insertion_dk(char flag,long k)
{
char flag_;

if(flag==0){                        /* ptmp, dk */
flag_=pdata_increase(k,&ptmp[0],dk);
}/**if(flag)**/
else if(flag==1){                   /* ptmp_line, dk_line */
flag_=pdata_increase(k,&ptmp_line[0],dk_line);
}/**else if(flag)**/
else{                               /* ptmp_word, dk_word */
flag_=pdata_increase(k,&ptmp_word[0],dk_word);
}/**else(flag)**/

page_firstk(firstk);
```

```
if(flag_==0) {if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;}
}/** insertion_dk **/


void overwrite(void)
{
if(insorover==0) return;

overwriteflag=1;

if(dialogflag>0){
lumpflag_dialog=1;
deletion_dialog();
lumpflag_dialog=0;
}
else
deletion_onlymem();

overwriteflag=0;
}/** overwrite **/


int deletion_onlymem(void)
{
char type;
long k,dk;

tailcheck();

k=top_icsr(/*firstline+*/jcsr,icsr);
if(k==kmax[fn]) return 1;
if(overwriteflag==1){
if(p[fn][k]=='\n') return 1;
}

type=gettype_p(k);

if(type<=2){
/*to_stack(p[fn][k]);*/
dk=1;
memmove(&p[fn][k],&p[fn][k+dk],kmax[fn]-(k+dk)+1);
kmax[fn]-=dk;
}/**if(type)**/
else if(type==3){
/*to_stack_2b(p[fn][k],p[fn][k+1]);*/
dk=2;
```

```
memmove(&p[fn][k],&p[fn][k+dk],kmax[fn]-(k+dk)+1);
kmax[fn]-=dk;
}/**else if(type)**/
else{
}/**else(type)**/

if(overwriteflag==1){
if(cut>0 && k<k_from) k_from+=-dk;  /* < */
}

return 0;
}/** deletion_onlymem **/


int deletion(void)
{
char type;
long k,dk;

tailcheck();

k=top_icsr(/*firstline+*/jcsr,icsr);
if(k==kmax[fn]) return 1;

type=gettype_p(k);

if(type<=2){
to_stack(p[fn][k]);
dk=1;
memmove(&p[fn][k],&p[fn][k+dk],kmax[fn]-(k+dk)+1);
kmax[fn]-=dk;
}/**if(type)**/
else if(type==3){
to_stack_2b(p[fn][k],p[fn][k+1]);
dk=2;
memmove(&p[fn][k],&p[fn][k+dk],kmax[fn]-(k+dk)+1);
kmax[fn]-=dk;
}/**else if(type)**/
else{
}/**else(type)**/

if(cut>0 && k<k_from) k_from+=-dk;  /* < */

page_firstk(firstk);

if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;
```

```
return 0;
}/** deletion **/


void deletion_dk_lump(void)
{
memmove(&p[fn][k_from_rep],&p[fn][k_to_rep],kmax[fn]-k_to_rep+1);
kmax[fn]-=k_to_rep-k_from_rep;
}/** deletion_dk_lump **/


void deletion_dk(void)
{
memmove(&p[fn][k_from],&p[fn][k_to],kmax[fn]-k_to+1);
kmax[fn]-=k_to-k_from;

page_firstk_from();

if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;
}/** deletion_dk **/


void page_firstk_from(void)
{
firstk=firstk_from;                  /* 3vals */
icsr=icsr_from;jcsr=jcsr_from;
page_firstk(firstk);

within_linemax();
}/** page_firstk_from **/


void delay_(long millisecond)
{
long oldtime,nowtime,dtime;
double i=CLOCKS_PER_SEC,j;

j=millisecond;
millisecond=j*(i/1000.);
oldtime=clock();

while(1){
nowtime=clock();dtime=nowtime-oldtime;
if(dtime>=millisecond) break;
if(dtime<0) break;
```

```
}
}/** delay_ **/


void beep(long millisecond)
{
/*unsigned cnt;

cnt=1193280/555;

ioperm(0x43,1,1);
ioperm(0x42,1,1);
ioperm(0x61,1,1);

outb(0xb6,0x43);
outb(cnt & 0xff,0x42);
outb((cnt & 0xff00) >> 8,0x42);

outb(inb(0x61) | 0x03,0x61);
delay_(millisecond);
outb(inb(0x61) & 0xfc,0x61);

ioperm(0x43,1,0);
ioperm(0x42,1,0);
ioperm(0x61,1,0);*/
}/** beep **/


int getTAB(int i)
{
/*return TAB_c;*/
return TAB_c-i%TAB_c;
}/** getTAB **/


void csr_tab_dialog(char leftorright)
{
char flag_tab,flag_cc,flag_2b,type;
int icsr_,ris;
long k;

k=firstk_dialog;
icsr_=0;
flag_tab=0;
flag_cc=0;
flag_2b=0;
```

```
if(icsr==0){
}/**if(icsr)**/
else{
while(1){
type=gettype_dialog(k);

if(type<=2){
k++;

if(type<=0){
icsr_++;
if(icsr_==icsr) {/*flag=0;*/break;}
}/**if(type)**/
else if(type==1){                    /* Tab */
icsr_++;
if(icsr_==icsr) {flag_tab=0;break;}
if(icsr_>icsr) {flag_tab=1;break;}
}/**else if(type)**/
else{                                /* Control code */
icsr_++;
if(icsr_==icsr) {flag_cc=0;break;}
if(icsr_>icsr) {flag_cc=1;break;}
}/**else(type)**/
}/**if(type)**/
else if(type==3){
k+=2;

icsr_+=2;
if(icsr_==icsr) {flag_2b=0;break;}
if(icsr_>icsr) {flag_2b=1;break;}
}/**else if(type)**/
else{
}/**else(type)**/
}/**while(1)**/

if(leftorright==0)
icsr=icsr_-flag_2b*2;
else{
icsr=icsr_;
}
}/**else(icsr)**/
}/** csr_tab_dialog **/


void csr_tab(char leftorright)
```

```
{
char flag_tab,flag_cc,flag_2b,type;
int icsr_,ris,TAB;
long k,line;

line=/*firstline+*/jcsr;
k=topp[line];
icsr_=0;
flag_tab=0;
flag_cc=0;
flag_2b=0;

if(icsr==0){
}/**if(icsr)**/
else{
while(1){
type=gettype_p(k);

if(type<=2){
k++;

if(type<=0){
icsr_++;
if(icsr_==icsr) {/*flag=0;*/break;}
}/**if(type)**/
else if(type==1){                    /* Tab */
TAB=getTAB(icsr_);                   /* TAB_v */

icsr_+=TAB;
if(icsr_==icsr) {flag_tab=0;break;}
if(icsr_>icsr) {flag_tab=1;break;}
}/**else if(type)**/
else{                                /* Control code */
icsr_+=2;
if(icsr_==icsr) {flag_cc=0;break;}
if(icsr_>icsr) {flag_cc=1;break;}
}/**else(type)**/
}/**if(type)**/
else if(type==3){
k+=2;

icsr_+=2;
if(icsr_==icsr) {flag_2b=0;break;}
if(icsr_>icsr) {flag_2b=1;break;}
}/**else if(type)**/
else{
```

```
}/**else(type)**/
}/**while(1)**/

if(leftorright==0)
icsr=icsr_-flag_tab*TAB-flag_cc*2-flag_2b*2;
else{
icsr=icsr_;
ris=return_is(line);
if(icsr>ris) {icsr=ris;csr_right();}  /* over COLUMN-1 : Tab, Control code, 2b */
}
}/**else(icsr)**/
}/** csr_tab **/


long top_icsr(int jcsr,int icsr_auto)
{
char flag_tab,flag_cc,flag_2b,type;
int icsr_,ris,TAB;
long k,dk_auto;

k=topp[jcsr];
icsr_=0;
flag_tab=0;
flag_cc=0;
flag_2b=0;
flag_2nd=0;

if(icsr_auto==0){
}/**if(icsr_auto)**/
else{
while(1){
type=gettype_p(k);

if(type<=2){
dk_auto=1;
k+=dk_auto;

if(type<=0){
icsr_++;
if(icsr_==icsr_auto) {/*flag=0;*/break;}
}/**if(type)**/
else if(type==1){                   /* Tab */
TAB=getTAB(icsr_);                  /* TAB_v */

icsr_+=TAB;
if(icsr_==icsr_auto) {flag_tab=0;break;}
```

```
if(icsr_>icsr_auto) {flag_tab=1;k-=dk_auto;break;}
}/**else if(type)**/
else{                                /* Control code */
icsr_+=2;
if(icsr_==icsr_auto) {flag_cc=0;break;}
if(icsr_>icsr_auto) {flag_cc=1;k-=dk_auto;break;}
}/**else(type)**/
}/**if(type)**/
else if(type==3){
dk_auto=2;
k+=dk_auto;

icsr_+=2;
if(icsr_==icsr_auto) {flag_2b=0;break;}
if(icsr_>icsr_auto) {flag_2b=1;k-=dk_auto;flag_2nd=1;break;}
}/**else if(type)**/
else{
}/**else(type)**/
}/**while(1)**/
}/**else(icsr_auto)**/

return k;
}/** top_icsr **/



int return_is(long jcsr)
{
char type,breakflag;
int i,TAB;
long k;
unsigned char s[1];

k=topp[jcsr];
i=0;

while(1){
s[0]=p[fn][k];
type=gettype_p(k);

if(type<=2){
if(type<=0) i++;
else if(type==1) {TAB=getTAB(i);i+=TAB;}  /* TAB_v */
else i+=2;

if(s[0]=='\n') break;
else{
```

```
/*if(i>COLUMN-1) {i=COLUMN;break;}*/

if(type<=0 && i==COLUMN) breakflag=1;
else if(type==1 && i>COLUMN-1) breakflag=1;
else if(type==2 && i>COLUMN-1) breakflag=1;
else  breakflag=0;

if(breakflag==1) {i=COLUMN;break;}
}

/*if(k>=kmax[fn]) break;*/
k++;

if(k>kmax[fn]) break;               /* new break */
}/**if(type)**/
else if(type==3){
i+=2;

if(i>=COLUMN) {i=COLUMN;break;}

/*if(k>=kmax[fn]) break;*/              /* ? */
k+=2;

if(k>kmax[fn]) break;               /* new break */
}/**else if(type)**/
else{
}/**else(type)**/
}/**while(1)**/

return i-1;
}/** return_is **/


long getspan_u(void)
{
char type;
long k,dk_auto;

k=0;dk_auto=0;

while(1){
if(k==delsp) break;

type=gettype_u(k);

if(type<=2) {dk_auto=1;k+=dk_auto;}
```

```
else if(type==3) {dk_auto=2;k+=dk_auto;}
else{}
}/**while(1)**/

return dk_auto;
}/** getspan_u **/


int kinsoku(/*unsigned char *str,*/long k/*,long kmax*/)
{
char type;

if(Flag_k==0) return 0;
if(k==kmax[fn]) return 0;

type=gettype_p(/*str,*/k/*,kmax*/);

if(type<=2){
if(p[fn][k]==',' || p[fn][k]=='.') {/*beep(100);*/return 1;}
else if(p[fn][k]==')' || p[fn][k]=='}' || p[fn][k]==']') return 1;
else if(p[fn][k]=='_' || p[fn][k]=='^') return 1;
else if(p[fn][k]=='?' || p[fn][k]=='!') return 1;
else if(p[fn][k]=='\'') return 1;
/*else if(k-7>=0 && strncmp(&p[fn][k-7],"\aDELTA\b",7)==0) return 1;*/
}
else if(type==3){
                                    /* EUC (1st:JIS+0x80, 2nd:JIS+0x80) */
if(p[fn][k]==0xa1 && (p[fn][k+1]==0xa2 || p[fn][k+1]==0xa3)) return 1;
else if(p[fn][k]==0xa1 && (p[fn][k+1]==0xd7 || p[fn][k+1]==0xd9)) return 1;
}

return 0;
}/** kinsoku **/


char gettype_jp(long k)
{
char type;
int jcsr_ini;
int icsr_old,jcsr_old;
long firstk_old;
long member;

if(k==kmax[fn]) return -1;

member=k;
```

```
type=gettype_p(member);

if(type!=0 ||
   (p[fn][member]<0x30 && p[fn][member]!=0x24) ||
   (p[fn][member]>0x39 && p[fn][member]<0x41) ||
   (p[fn][member]>0x5A && p[fn][member]<0x5F) ||
    p[fn][member]>0x7A || p[fn][member]==0x60
)
{                                   /* control code, symbol(1byte), 2bytes */
if(
                                                                      /* kanji */
type==3 && (p[fn][member]>=0xb0 || (p[fn][member]==0xa1 && p[fn][member+1]==0xa8))
)
/*wordcheck_2bytes(0,operation,k);*/return 3;
else if(
type==3 && ((p[fn][member]==0xa5 && p[fn][member+1]<=0xf6)  /* katakana */
            || (p[fn][member]==0xa1 && p[fn][member+1]==0xbc))
)
/*wordcheck_2bytes(1,operation,k);*/return 4;
else if(
type==3 && p[fn][member]==0xa4 && p[fn][member+1]>=0xa1  /* hiragana */
)
/*wordcheck_2bytes(2,operation,k);*/return 5;
else if(
type==3 && p[fn][member]==0xa3 && p[fn][member+1]<=0xfa  /* alphabet, figure */
)
/*wordcheck_2bytes(3,operation,k);*/return 6;
else if(
type==3 && (p[fn][member]==0xa6)    /* greek */
)
/*wordcheck_2bytes(4,operation,k);*/return 7;
else if(
/*tabspaces==1 && (*/(type==1) ||       /* tab, half space, full space */
(type==0 && p[fn][member]==0x20) ||
(type==3 && p[fn][member]==/*0xa1*/SPC1 && p[fn][member+1]==/*0xa1*/SPC2)/*)*/
)
/*wordcheck_unvisible(operation,k);*/return 8;
else if(
type==3 && (p[fn][member]==0xa1 || p[fn][member]==0xa2)  /* symbol(2bytes) */
)
return 9;
else if(
type==3 && (p[fn][member]>=0xa7 && p[fn][member]<=0xad)  /* symbol(2bytes) */
)
return 9;
else{                               /* control code, symbol(1byte) */
```

```
/*k_to=k_from=member;
if(operation==0) {if(lumpflag_global==0) lumpflag=0;deletion();}*/
return 1;
}

/*return 1;*/
}/**if(type,p[fn][member])**/

return 0;                              /* word */
}/** gettype_jp **/


char gettype(char flag,unsigned char s1/*,unsigned char s2*/,long k,long kend)
{
char type;

/*#ifdef UNICODE
WCHAR s1=s1_;
#else
unsigned char s1=s1_;
#endif*/

/*#ifdef UNICODE
if(s1>=0x20 && s1<=0x7e) type=0;
else if(s1==0x203e) type=0;
else if(s1>=0xff61 && s1<=0xff9f) type=0;
else if(flag==1 && k==kend) type=0;
else if(s1==0x0a) type=0;
else if(s1==0x09) type=1;
else if((s1>0x00 && s1<0x20) || s1==0x7f) type=2;
else if(s1==0x00) type=-1;
else type=3;

#else*/

/* EUC-> */
if(s1>=0x20 && s1<=0x7e) type=0;                    /* word */
/*else if(s1>=0xa1 && s1<=0xdf) type=0;*/               /* kana(1byte) */
/*else if(flag==0 && s2=='\0') type=0;*/            /* ? */
else if(flag==1 && k==kend) type=0;                 /* end of string */
else if(s1==0x0a) type=0;                           /* control code(LF) */
else if(s1==0x09) type=1;                           /* control code(HT) */
else if((s1>0x00 && s1<0x20) || s1==0x7f) type=2;    /* control code(others) */
else if(s1==0x00) type=-1;                          /* '\0' */
/*else if(s1<=0xff) type=0;*/                           /* word(option) */
else type=3;                                        /* Double Byte Character */
```

```
/* <-EUC */
/*else if(s1>=0xa1 && s1<=0xfe && s2>=0xa1 && s2<=0xfe) type=3;
else if(s1==0x8f && s2>0x80) type=3;*/               /* Double Byte Character */
/*#endif*/

return type;
}/** gettype **/


char gettype_buf(long k,unsigned char *buf)
{
char type;
unsigned char s[2];

s[0]=buf[k];
/*s[1]=buf[k+1];*/

type=gettype(0,s[0]/*,s[1]*/,k,-1);

return type;
}/** gettype_buf **/


char gettype_fnames(int j,long k)
{
char type;
unsigned char s[2];

s[0]=fnames[ftable[j-1].fn][k];
/*s[1]=fnames[ftable[j-1].fn][k+1];*/

type=gettype(0,s[0]/*,s[1]*/,k,-1);

return type;
}/** gettype_fnames **/


char gettype_ac(long k)
{
char type;
unsigned char s[2];

s[0]=array[k];
/*s[1]=array[k+1];*/

type=gettype(0,s[0]/*,s[1]*/,k,-1);
```

```
return type;
}/** gettype_ac **/


char gettype_mline(long k)
{
char type;
unsigned char s[2];

s[0]=mline[k];
/*s[1]=mline[k+1];*/

type=gettype(1,s[0]/*,s[1]*/,k,kmax_ml);

return type;
}/** gettype_mline **/


char gettype_u(long k)
{
char type;
unsigned char s[2];

s[0]=stack_del[k];
/*s[1]=stack_del[k+1];*/

type=gettype(1,s[0]/*,s[1]*/,k,delsp-1);

return type;
}/** gettype_u **/


char gettype_dialog(long k)
{
char type;
unsigned char s[2];

s[0]=p_dialog[k];
/*s[1]=p_dialog[k+1];*/

type=gettype(1,s[0]/*,s[1]*/,k,kmax_dialog);

return type;
}/** gettype_dialog **/
```

```
char gettype_p(long k)
{
char type;
unsigned char s[2];

s[0]=p[fn][k];
/*if(k+1<=kmax[fn])
s[1]=p[fn][k+1];*/

type=gettype(1,s[0]/*,s[1]*/,k,kmax[fn]);

/*if(s[0]>=0x20 && s[0]<0x7f) type=0;
else if(s[0]>=0xa1 && s[0]<=0xdf) type=0;
else if(s[0]==0x0a || k==kmax[fn]) type=0;
else if(s[0]==0x09) type=1;
else if((s[0]>0x00 && s[0]<0x20) || s[0]==0x7f) type=2;
else if(s[0]>=0xa1 && s[0]<=0xfe && s[1]>=0xa1 && s[1]<=0xfe) type=3;
else if(s[0]==0x8f && s[1]>0x80) type=3;
else type=-1;*/

return type;
}/** gettype_p **/


char gettype_sdb(long k)
{
char type;
unsigned char s[2];

s[0]=stock_db[k];
/*s[1]=stock_db[k+1];*/

type=gettype(1,s[0]/*,s[1]*/,k,kmax_sdb);

return type;
}/** gettype_sdb **/


void clear_topp(void)
{
int i=0;

while(1){
topp[i]=-1;
if(i==ROW) break;
```

```
i++;
}
}/** clear_topp **/


void while_puts_show_(char TextOutflag,long k)
{
char tabflag,ccflag,type,ijlineflag,onceflag;
int i,j,dx,dy,itab,icc,TAB;
unsigned char s[1],s_[1];
unsigned char jis[2];

clear_topp();

i=0;j=0;
onceflag=0;
tabflag=0;itab=0;                   /* Tab */
ccflag=0;icc=0;                     /* Control code */
topp[j]=k;

while(1){
s[0]=p[fn][k];
type=gettype_p(k);

if(type<=2){
if(type==1 && onceflag==0) {onceflag=1;TAB=getTAB(i);tabflag=1;}  /* TAB_v */
if(type==2) ccflag=1;               /* Control code */



if(TextOutflag){
if(s[0]>=0x20 && type==0)
setstccolor(bfset[WB].fore);
else if(type==-1)
setstccolor(12);
/*else if(s[0]==0x1a)*/
else if(k==kmax[fn])
setstccolor(12);
else if(s[0]=='\n')
setstccolor(RETURN);
else if(s[0]==0x09)
setstccolor(TABCOLOR);
else{
if(s[0]==18 || s[0]==14 || s[0]==25){
if(icc==0) setstccolor(/*CC*/9);else setstccolor(/*bfset[WB].fore*/9);
```

```
}
else if(s[0]==27 || s[0]==29){
if(icc==0) setstccolor(/*CC*/13);else setstccolor(/*bfset[WB].fore*/13);
}
else{
if(icc==0) setstccolor(/*CC*/12);else setstccolor(/*bfset[WB].fore*/12);
}
}

dx=(i+DI)*UDX;dy=(j+DJ)*UDY;

if(s[0]>=0x20 && type==0)
stc(1,dx,dy,s,1);
else if(s[0]=='\n'){
s_[0]=dummy_R;
stc(1,dx,dy,s_,1);
}
/*else if(s[0]==0x1a){*/
else if(k==kmax[fn]){
s_[0]=/*0x0d*/dummy_E;
stc(1,dx,dy,s_,1);
}
else if(type==-1){
s_[0]=0x0d;
stc(1,dx,dy,s_,1);
}
else if(s[0]==0x09){
s_[0]=dummy_T;
stc(1,dx,dy,s_,1);
}
else{
if(s[0]==0x7f) s[0]=0x00;
if(icc==0) s_[0]='^';
else s_[0]=cc[s[0]];
stc(1,dx,dy,s_,1);
}
}/**if(TextOutflag)**/

/*if(tabflag==0 && ccflag==0 && k>=kmax[fn]) break;*/    /* Tab, Control code */

if(tabflag==1){                     /* Tab */
itab++;
if(itab==TAB) {onceflag=0;tabflag=0;itab=0;k++;}
}
else if(ccflag==1){                 /* Control code */
icc++;
```

```
if(icc==2) {ccflag=0;icc=0;k++;}
}
else k++;



if(tabflag==0 && ccflag==0 && k>kmax[fn]) break;    /* new break */



i++;

if(s[0]=='\n'){
ijlineflag=/*1*/2;
}/**if(s[0])**/
else{
if(tabflag==0 && ccflag==0 && i==COLUMN){
ijlineflag=1;
}
else if(tabflag==1 /*&& ccflag==0 */&& i==COLUMN){  /* Tab */
onceflag=0;tabflag=0;itab=0;k++;
ijlineflag=1;
}
else if(/*tabflag==0 && ccflag==0 && */i==COLUMN+1){  /* Control code */
ijlineflag=1;
}
else{
ijlineflag=0;
}
}/**else(s[0])**/
}/**if(type)**/
else if(type==3){
if(TextOutflag){
jis[0]=p[fn][k];
jis[1]=p[fn][k+1];

dx=(i+DI)*UDX;dy=(j+DJ)*UDY;

setstccolor(bfset[WB].fore);
stc(1,dx,dy,jis,2);
}/**if(TextOutflag)**/

/*if(k>=kmax[fn]) break;*/              /* ? */

k+=2;
```

```
if(k>kmax[fn]) break;               /* new break */



i+=2;

if(i>=COLUMN) ijlineflag=1;
else          ijlineflag=0;
}/**else if(type)**/
else{
}/**else(type)**/

if(ijlineflag==1){
if(kinsoku(k)==1) ijlineflag=0;
}

if(ijlineflag>0){
i=0;j++;
topp[j]=k;

#if GRP_or_EDT==0
break;
#endif
}

if(j==ROW+1) break;
}

jcsrmax=min(j,ROW);

#if GRP_or_EDT==0
k_g=k;if(k_g>kmax[fn]) k_g=kmax[fn];
#endif
}/** while_puts_show_ **/



int while_puts_thepart(long k_left,long k_right)
{
char TextOutflag,tabflag,ccflag,type,ijlineflag,onceflag;
int i,j,dx,dy,itab,icc,TAB;
long k,line;
unsigned char s[1];

TextOutflag=0;
```

```
i=0;j=0;
onceflag=0;
tabflag=0;itab=0;                    /* Tab */
ccflag=0;icc=0;                      /* Control code */
k=k_left;line=0;

while(1){
s[0]=p[fn][k];
type=gettype_p(k);

if(type<=2){                         /* single byte */
if(type==1 && onceflag==0) {onceflag=1;TAB=getTAB(i);tabflag=1;}  /* TAB_v */
if(type==2) ccflag=1;                /* Control code */

if(TextOutflag){
}/**if(TextOutflag)**/

if(k==k_right+1) break;              /* special */

/*if(tabflag==0 && ccflag==0 && k>=kmax[fn]) break;*/    /* Tab, Control code */

/*k++;*/
if(tabflag==1){                      /* Tab */
itab++;
if(itab==TAB) {onceflag=0;tabflag=0;itab=0;k++;}
}
else if(ccflag==1){                  /* Control code */
icc++;
if(icc==2) {ccflag=0;icc=0;k++;}
}
else k++;



if(tabflag==0 && ccflag==0 && k>kmax[fn]) break;    /* new break */



i++;

#if 0
if(s[0]=='\n'){
ijlineflag=/*1*/2;
}/**if(s[0])**/
else{
if(tabflag==0 && ccflag==0 && i==COLUMN){
```

```
ijlineflag=1;
}
else if(tabflag==1 /*&& ccflag==0 */&& i==COLUMN){  /* Tab */
onceflag=0;tabflag=0;itab=0;k++;
ijlineflag=1;
}
else if(/*tabflag==0 && ccflag==0 && */i==COLUMN+1){  /* Control code */
ijlineflag=1;
}
else{
ijlineflag=0;
}
}/**else(s[0])**/
#endif
}/**if(type)**/
else if(type==3){                   /* double byte */
if(TextOutflag){
}/**if(TextOutflag)**/

if(k==k_right+1) break;             /* special */

/*if(k>=kmax[fn]) break;*/              /* ? */

k+=2;



if(k>kmax[fn]) break;               /* new break */



i+=2;

#if 0
if(i>=COLUMN) ijlineflag=1;
else          ijlineflag=0;
#endif
}/**else if(type)**/
else{
}/**else(type)**/

/*if(j==ROW) break;*/
}

return i;
}/** while_puts_thepart **/
```

```
long while_puts_firstk(long kstart,long line_firstk)
{
char TextOutflag,tabflag,ccflag,type,ijlineflag,onceflag;
int i,j,dx,dy,itab,icc,TAB;
long k,line;
unsigned char s[1];

TextOutflag=0;
i=0;j=0;
onceflag=0;
tabflag=0;itab=0;                   /* Tab */
ccflag=0;icc=0;                     /* Control code */
k=kstart;line=0;
if(line==line_firstk) {line_end=line;return k;}

while(1){
s[0]=p[fn][k];
type=gettype_p(k);

if(type<=2){                        /* single byte */
if(type==1 && onceflag==0) {onceflag=1;TAB=getTAB(i);tabflag=1;}  /* TAB_v */
if(type==2) ccflag=1;               /* Control code */

if(TextOutflag){
}/**if(TextOutflag)**/

/*if(tabflag==0 && ccflag==0 && k>=kmax[fn]) break;*/    /* Tab, Control code */

/*k++;*/
if(tabflag==1){                     /* Tab */
itab++;
if(itab==TAB) {onceflag=0;tabflag=0;itab=0;k++;}
}
else if(ccflag==1){                 /* Control code */
icc++;
if(icc==2) {ccflag=0;icc=0;k++;}
}
else k++;



if(tabflag==0 && ccflag==0 && k>kmax[fn]) break;    /* new break */
```

```
i++;

if(s[0]=='\n'){
ijlineflag=1;
}/**if(s[0])**/
else{
if(tabflag==0 && ccflag==0 && i==COLUMN){
ijlineflag=1;}
else if(tabflag==1 /*&& ccflag==0 */&& i==COLUMN){  /* Tab */
onceflag=0;tabflag=0;itab=0;k++;
ijlineflag=1;}
else if(/*tabflag==0 && ccflag==0 && */i==COLUMN+1){  /* Control code */
ijlineflag=1;}
else ijlineflag=0;
}/**else(s[0])**/
}/**if(type)**/
else if(type==3){                   /* double byte */
if(TextOutflag){
}/**if(TextOutflag)**/

/*if(k>=kmax[fn]) break;*/              /* ? */

k+=2;



if(k>kmax[fn]) break;               /* new break */



i+=2;

if(/*i==COLUMN || i==COLUMN+1*/i>=COLUMN)
ijlineflag=1;
else ijlineflag=0;
}/**else if(type)**/
else{
}/**else(type)**/

if(ijlineflag==1){
i=0;j++;
/*if((type<=2 && s[0]=='\n')||(jumpflag==1 && linelength==0)||jumpflag==0)*/
if(linelength_new==0) line++;  /* inside work */
else{
if(LINEMODE==0) line++;
```

```
else {if(type<=2 && s[0]=='\n') line++;}
}

if(line==line_firstk) break;
}

/*if(j==ROW) break;*/
}

line_end=line;

return k;
}/** while_puts_firstk **/


long while_puts_dline(long kstart,long kend)
{
char TextOutflag,tabflag,ccflag,type,ijlineflag,onceflag;
int i,j,dx,dy,itab,icc,TAB;
long k,line;
unsigned char s[1];

TextOutflag=0;
i=0;j=0;
onceflag=0;
tabflag=0;itab=0;                   /* Tab */
ccflag=0;icc=0;                     /* Control code */
k=kstart;line=0;

while(1){
s[0]=p[fn][k];
type=gettype_p(k);

if(type<=2){                        /* single byte */
if(type==1 && onceflag==0) {onceflag=1;TAB=getTAB(i);tabflag=1;}  /* TAB_v */
if(type==2) ccflag=1;               /* Control code */

if(TextOutflag){
}/**if(TextOutflag)**/

if(function>0 && k==kend) icsr_global=i;
if(function==2 && k==kend) icsr_last=i;
if(k==kend) break;                  /* special */

/*if(tabflag==0 && ccflag==0 && k>=kmax[fn]) break;*/    /* Tab, Control code */
```

```
/*k++;*/
if(tabflag==1){                     /* Tab */
itab++;
if(itab==TAB) {onceflag=0;tabflag=0;itab=0;k++;}
}
else if(ccflag==1){                 /* Control code */
icc++;
if(icc==2) {ccflag=0;icc=0;k++;}
}
else k++;



if(tabflag==0 && ccflag==0 && k>kmax[fn]) break;    /* new break */



i++;

if(s[0]=='\n'){
ijlineflag=1;
}/**if(s[0])**/
else{
if(tabflag==0 && ccflag==0 && i==COLUMN){
ijlineflag=1;}
else if(tabflag==1 /*&& ccflag==0 */&& i==COLUMN){  /* Tab */
onceflag=0;tabflag=0;itab=0;k++;
ijlineflag=1;}
else if(/*tabflag==0 && ccflag==0 && */i==COLUMN+1){  /* Control code */
ijlineflag=1;}
else ijlineflag=0;
}/**else(s[0])**/
}/**if(type)**/
else if(type==3){                   /* double byte */
if(TextOutflag){
}/**if(TextOutflag)**/

/*if(function>0 && k==kend) icsr_global=i;*/
if(function>0 && (k==kend-1 || k==kend)) icsr_global=i;
if(function==2 && (k==kend-1 || k==kend)) icsr_last=i;
if(k==kend-1 || k==kend) break;     /* special */

/*if(k>=kmax[fn]) break;*/              /* ? */

k+=2;
```

```
if(k>kmax[fn]) break;               /* new dialog */



i+=2;

if(/*i==COLUMN || i==COLUMN+1*/i>COLUMN-1)
ijlineflag=1;
else ijlineflag=0;
}/**else if(type)**/
else{
}/**else(type)**/

if(ijlineflag==1){
i=0;j++;
/*if((type<=2 && s[0]=='\n')||(jumpflag==1 && linelength==0)||jumpflag==0)*/
if(linelength_new==0) line++;  /* inside work */
else{
if(LINEMODE==0) line++;
else {if(type<=2 && s[0]=='\n') line++;}
}
}

/*if(j==ROW) break;*/
}

return line;
}/** while_puts_dline **/


void while_puts_theline(int jcsr_ini)
{
long ddline;

ddline=get_firstk(member_global,jcsr_ini);
if(ddline<0) jcsr=ddline+jcsr_ini;
else jcsr=jcsr_ini;

icsr=icsr_global;
}/** while_puts_theline **/


void while_puts_fload_(char flag,int jcsr_ini)
{
```

```
long ddline;

if(flag==0) while_puts_theline(jcsr_ini);
else{
ddline=get_firstk(member_last,jcsr_ini);
if(ddline<0) jcsr=ddline+jcsr_ini;
else jcsr=jcsr_ini;

icsr=icsr_last;                     /* <- function = 2 */
}
}/** while_puts_fload_ **/


void page_firstk(long k)
{
firstk=max(k,0);if(firstk>kmax[fn]) get_firstk(kmax[fn],0);  /* protection */

if(lumpflag>0) {while_puts_show_(0,firstk);return;}

cleardevice_(-1,0,0,0,0);
while_puts_show_(1,firstk);      /* 1 : TextOut to plane_1 */

BitBlt_full();
/*printf_((int)firstk);use_subroop();*/
}/** page_firstk **/


void text_home(void)
{
get_firstk(0,0);
page_firstk(firstk);

csr_column_home();csr_row_home();
}/** text_home **/


void text_end(void)
{
get_firstk(kmax[fn],ROW-1);
page_firstk(firstk);

csr_column_end();csr_row_end();
}/** text_end **/


void page_down(void)
```

```
{
if(filerflag==0){
while_puts_firstk(firstk,/*ROW-1*/ROW+jcsr);
if(/*ROW-1*/ROW+jcsr==line_end){
/*printf_((int)firstk);use_subroop();*/
page_firstk(topp[/*ROW-1*/ROW]);
/*printf_((int)firstk);use_subroop();*/
}
else{
get_firstk(kmax[fn],jcsr);
page_firstk(firstk);
}
}/**if(filerflag)**/
else{
while_puts_firstk(firstk,/*ROW-1*/ROW+jcsr+1);  /* jcsr+1 */
if(/*ROW-1*/ROW+jcsr+1==line_end){
page_firstk(topp[/*ROW-1*/ROW]);
}
else{
get_firstk(kmax[fn],jcsr+1);
page_firstk(firstk);
}
}/**else(filerflag)**/
}/** page_down **/


void page_up(void)
{
/*printf_((int)firstk);use_subroop();*/
get_firstk(firstk,/*ROW-1*/ROW);
page_firstk(firstk);
}/** page_up **/


int scroll_down(char moveflag)
{
if(filerflag==0){
/*while_puts_firstk(firstk,1);
if(line_end==0)  return 1;*/
if(topp[1]==-1) return 1;
}
else{
/*while_puts_firstk(firstk,2);
if(line_end==1)  return 1;*/
if(topp[2]==-1) return 1;
}
```

```
if(moveflag==1){
if(jcsr>0) jcsr--;
else jcsr=0;
}
else{
if(jcsr==jcsrmax) jcsr--;
}

page_firstk(topp[1]);

return 0;
}/** scroll_down **/


int scroll_up(char moveflag)
{
if(topp[0]==0) return 1;

if(moveflag==1){
if(jcsr<ROW-1) jcsr++;
else jcsr=ROW-1;
}

get_firstk(firstk,1);
page_firstk(firstk);

return 0;
}/** scroll_up **/


void within_linemax(void)
{
if(/*firstline+*/jcsr>jcsrmax) jcsr=jcsrmax;
}/** within_linemax **/


void csr_column_home(void)
{
jcsr=0;
}/** csr_column_home **/


void csr_column_end(void)
{
jcsr=ROW-1;
```

```
within_linemax();
}/** csr_column_end **/


void csr_row_home(void)
{
within_linemax();

icsr=0;
}/** csr_row_home **/


void csr_row_end(void)
{
int ris;

within_linemax();

ris=return_is(/*firstline+*/jcsr);
icsr=ris;
}/** csr_row_end **/


void csr_down(void)
{
jcsr++;
within_linemax();
if(jcsr>ROW-1) {jcsr=ROW-1;scroll_down(0);}
}/** csr_down **/


void csr_up(void)
{
jcsr--;
within_linemax();
if(jcsr<0) {jcsr=0;scroll_up(0);}
}/** csr_up **/


int csr_left(void)
{
int ris;
long k;

if(/*firstline+*/jcsr>jcsrmax){
```

```
jcsr=jcsrmax;
ris=return_is(/*firstline+*/jcsr);
icsr=ris;
}/**if(firstline,jcsr)**/
else{
ris=return_is(/*firstline+*/jcsr);
if(icsr>ris){
icsr=ris;
if(ris==COLUMN-1){                  /* <-> if(flag_2nd) icsr--, but ? */
k=top_icsr(/*firstline+*/jcsr,icsr);
if(flag_2nd==0 && gettype_p(k)==3) icsr--;
}
}
else{
/*k=*/top_icsr(/*firstline+*/jcsr,icsr);if(flag_2nd) icsr--;
icsr--;
}

if(icsr<0){
jcsr--;

if(jcsr<0){
jcsr=0;

if(scroll_up(0)==1){
icsr=0;
return 1;}                          /* return 1 */
else{
icsr=return_is(/*firstline*/0);}

}/**if(jcsr)**/
else{
icsr=return_is(/*firstline+*/jcsr);
}/**else(jcsr)**/
}/**if(icsr)**/
}/**else(firstline,jcsr)**/

csr_tab(0);

return 0;                           /* return 0 */
}/** csr_left **/


void csr_right(void)
{
int ris;
```

```
if(/*firstline+*/jcsr>jcsrmax){
jcsr=jcsrmax;
ris=return_is(/*firstline+*/jcsr);
icsr=ris;
}/**if(firstline,jcsr)**/
else{
icsr++;
ris=return_is(/*firstline+*/jcsr);

if(icsr>ris){
jcsr++;                              /* -> */

if(jcsr>ROW-1 && topp[jcsr]!=-1){
jcsr=ROW-1;

if(scroll_down(0)==1){               /* impossible */
icsr=return_is(/*firstline+*/ROW-1);
}
else{
icsr=0;}

}/**if(jcsr)**/
else{
icsr=0;
}/**else(jcsr)**/
}/**if(icsr)**/

if(/*firstline+*/jcsr>jcsrmax){
jcsr--;                              /* <- */  /* or jcsr=jcsrmax; */
icsr=return_is(/*firstline+*/jcsr);}
}/**else(firstline,jcsr)**/

csr_tab(1);
}/** csr_right **/


int fload(char flag_rn)
{
char flag_;
long linefrom1;

flag_=0;

if(flag_rn!=2){
fseek(fp,0L,SEEK_END);
```

```
kmax[fn]=ftell(fp)-1;
kceil[fn]=kmax[fn]+1;                          /* +1 : for 0x1a */
p[fn]=(unsigned char *)malloc(kceil[fn]+(1+1)*sizeof(unsigned char));

if(p[fn]!=NULL){
fseek(fp,0L,SEEK_SET);
fread(p[fn],1,(kmax[fn]+1),fp);
kmax[fn]++;p[fn][kmax[fn]]=0x1a;

if(beginjumpflag){
linelength_new=1;
linefrom1=atol(linestring);if(linefrom1<1) linefrom1=1;
firstk=while_puts_firstk(0,linefrom1-1);
if(linefrom1-1!=line_end) get_firstk(kmax[fn],0);
linelength_new=0;
/*beginjumpflag=0;*/
}

if(flag_rn==1) {editflag[fn]=-1;/*strcpy(attri,"RO  ");*/}
else {editflag[fn]=0;/*strcpy(attri,"");*/}
#if GRP_or_EDT==1
page_firstk(firstk);
#endif
}/**if(p[fn])**/
else{
flag_=2;
}/**else(p[fn])**/

fclose(fp);
}/**if(flag_rn)**/
else{
kmax[fn]=-1;
kceil[fn]=kmax[fn]+1;                          /* +1 : for 0x1a */
p[fn]=(unsigned char *)malloc(kceil[fn]+(1+1)*sizeof(unsigned char));

if(p[fn]!=NULL){
kmax[fn]++;p[fn][kmax[fn]]=0x1a;

if(flag_rn==1) {editflag[fn]=-1;/*strcpy(attri,"RO  ");*/}
else {editflag[fn]=0;/*strcpy(attri,"");*/}
#if GRP_or_EDT==1
page_firstk(firstk);
#endif

if(access(fname,0)==0) puts_mline(0,"The file exists.");BitBltflag=1;
}/**if(p[fn])**/
```

```
else{
flag_=2;
}/**else(p[fn])**/
}/**else(flag_rn)**/

/*if(unlinkflag==0 && NKF==1) unlink(home_euc);*/
if(beginjumpflag) beginjumpflag=0;

if(flag_==0){
return 0;
}
else{
#if GRP_or_EDT==1
message(7,0);
#endif
return 1;
}
}/** fload **/


int fsave(char flag_bak,char flag_append)
{
char restore_fname;
int existence,writable,length;
unsigned char bak[ASIZE];

restore_fname=0;

existence=access(fname,0);
writable=access(fname,2);

if(existence==0 && writable==-1) {/*beep(500);*/return 1;}

if(existence==0){
strcpy(bak,fname);length=strlen(bak);
if(length<ASIZEM-1){
bak[length]='~';
bak[length+1]='\0';

CopyFile(fname,bak);
/*unlink(fname);*/
}
else if(length==ASIZEM-1){
strcpy(bak,home_global);
strcat(bak,"zzz.$$$");
```

```
CopyFile(fname,bak);
/*unlink(fname);*/
}
else{}                              /* impossible */
}

if(flag_append) openmode="ab";
else openmode="wb";

if((fp=fopen(fname,openmode))==NULL) restore_fname=1;
else{
if((int)fwrite(p[fn],1,kmax[fn]/*+1*/,fp)<kmax[fn]){  /* without 0x1a */
message(-ferror(fp),0);
clearerr(fp);
restore_fname=2;
}
fclose(fp);
}

if(restore_fname){
if(existence==0){
if(length<=ASIZEM-1){
if(restore_fname==2) CopyFile(bak,fname);
unlink(bak);
}
else{}                              /* impossible */
}/**if(existence)**/
else{
if(restore_fname==2) unlink(fname);
}/**else(existence)**/

/*if(restore_fname==1) */return 1;
/*else return 0;*/
}/**if(restore_fname)**/

if(existence==0 && flag_bak==0) unlink(bak);

if(nobeepflag==0){
editflag[fn]=0;
/*beep(50);*/
}

return 0;
}/** fsave **/
```

```
void mallocs(void)
{
int i;

ptmp_line=(unsigned char *)malloc(sizeof(unsigned char)*(COLUMN+1));
buf_line=(unsigned char *)malloc(sizeof(unsigned char)*(COLUMN+1));

p=(unsigned char **)malloc(sizeof(unsigned char *)*(FMAX+1));
editflag=(char *)calloc(FMAX+1,sizeof(char));
kmax=(long *)malloc(sizeof(long)*(FMAX+1));
kceil=(long *)malloc(sizeof(long)*(FMAX+1));
topp=(long *)malloc(sizeof(long)*(ROW_L+2));

fnames=(unsigned char **)malloc(sizeof(unsigned char *)*(FMAX+1));
i=0;
while(1){
fnames[i]=(unsigned char *)malloc(sizeof(unsigned char)*ASIZE);
i++;
if(i==FMAX) break;
}

fstack=(fs *)calloc(FMAX+1,sizeof(fs));
ftable=(ft *)malloc(sizeof(ft)*(FMAX+1));

kceiltmp=COLUMN-1;dk=kceiltmp-0+1;
ptmp=(unsigned char *)malloc(sizeof(unsigned char)*(kceiltmp+(1+1)));
}/** mallocs **/



void frees(void)
{
int i;

free(ptmp_line);
free(buf_line);

free(p);
free(editflag);
free(kmax);
free(kceil);
free(topp);

i=0;
while(1){
free(fnames[i]);
i++;
```

```
if(i==FMAX) break;
}
free(fnames);

free(fstack);
free(ftable);
}/** frees **/


void arrange_colors(void)
{
int bg,fg;

bg=bfset[WB].back;
fg=bfset[WB].fore;

if(fg==bg) {bg=15;fg=0;}
if(RTC==bg || RTC==fg) {bg=15;fg=0;RTC=9;}
if(ACTIVE==bg || INACTIVE==bg) {bg=15;fg=0;RTC=9;ACTIVE=13;INACTIVE=8;}
if(ACTIVE==RTC || INACTIVE==RTC) {bg=15;fg=0;RTC=9;ACTIVE=13;INACTIVE=8;}
if(CC==bg || CC==fg) CC=RTC;
}/** arrange_colors **/


int read_cfg(int cfgnumber)
{
int i,j,k;
int length;
/*unsigned */char buf[3],data[11];

/*itoa(cfgnumber+1,buf,10);*/gcvt(cfgnumber/*+1*/,3,buf);  /* +0 */
length=strlen(buf);

if(cfgmax+1<length+2+1) return -1000;

i=0;
while(1){
k=0;
while(1){
if(pcfg[i+k]==buf[k]) k++;
else break;
if(k==length) break;
}

if(k==length && pcfg[i+length]==':' && pcfg[i+length+1]==':'){
j=i+length+2;
```

```
while(1){
if((pcfg[j]==0x2d)||(pcfg[j]>=0x30 && pcfg[j]<=0x39)) j++;
else break;
if(j==cfgmax+1) break;
}

if(j==i+length+2) return -1000;
break;                                /* detected */
}/**if(k,pcfg[i+length],pcfg[i+length+1])**/

i++;
if(i==cfgmax-2-(length-1)) {/*beep(50);*/return -1000;}
}/**while**/

k=min(j-(i+length+2),/*10*/5);  /* <=5:five columns */
strncpy(data,&pcfg[i+length+2],k);
data[k]='\0';

/*printf(" %d\n",cfgnumber);*/

return atoi(data);
}/** read_cfg **/



void setup(void)
{
int XRESO_MAX,YRESO_MAX,maxfiles;
long av_mem,fsize;
unsigned char home[ASIZE];

strcpy(home_global,getenv("HOME"));
strcat(home_global,"/");
strcpy(home_global_GCD,home_global);

strcpy(home_deleted,home_global);
strcat(home_deleted,"zzz.deleted");
unlink(home_deleted);

strcpy(home_tmp,home_global);
strcat(home_tmp,"zzz.string");

strcpy(home_ref,home_global);
strcat(home_ref,"zzz.find");

if(0){
strcpy(fname_bg,home_global);
```

```
strcat(fname_bg,"cpage.bin");
}

two[0][0]=12;two[0][1]='\n';
two[1][0]=12;two[1][1]='\n';
two[2][0]=12;two[2][1]='\n';
two[3][0]=12;two[3][1]='\n';
two[4][0]='\n';

XRESO_MAX=DisplayWidth(d,screen);
YRESO_MAX=DisplayHeight(d,screen);

DX_FRAME=/*GetSystemMetrics(SM_CXSIZEFRAME)*/0;
DY_FRAME=/*GetSystemMetrics(SM_CYSIZEFRAME)*/0;
DY_CAPTION=/*GetSystemMetrics(SM_CYCAPTION)*/0;
DY_MENU=/*GetSystemMetrics(SM_CYMENU)*/0;
DY_TOOLBAR=0;

/*av_mem=get_av_memory(0,-1);*/

strcpy(home,home_global);
strcat(home,"ble.cfg");

if(GRP_or_EDT==0 || (fp=fopen(home,"rb"))==NULL){
start:

UDX=8;UDY=18;

     if(XRESO_MAX<=640)  {COLUMN=59;ROW_L=16;}
else if(XRESO_MAX<=800)  {COLUMN=79;ROW_L=22;}
else if(XRESO_MAX<=1024) {COLUMN=99;ROW_L=30;}
else if(XRESO_MAX<=1280){
if(YRESO_MAX<=800)       {COLUMN=99;ROW_L=30;}
else                     {COLUMN=119;ROW_L=30;}
}
else                     {COLUMN=119;ROW_L=30;}

RIGHT_M=3;

XRESO=(COLUMN+DI+DI_)*UDX+DX_FRAME*2+UDX*RIGHT_M;
YRESO=(ROW_L+/*2*/3)*UDY+(DY_CAPTION+DY_MENU+DY_TOOLBAR+DY_FRAME*2);

TAB_c=8;
CSRDY=UDY;
WB=0;
ACTIVE=13;
```

```
INACTIVE=8;
RTC=9;
RETURN=9;
TABCOLOR=3/*bfset[WB].back*/;
CC=9;
CSRCOLOR=15;
CSRCOLOR_FILER=12;
AINDENT=0;
fontnum=3;
dh=/*0*/-4;
dv=0;
l_s_flag=0;
tabspaces=1;
MOVEcsr=1;                          /* pm1 */
LEFT_m=60;
AVMEMDENO=8;
mfsize=/*av_mem/AVMEMDENO*/100;
FMAX=ROW_L+1;
}/**if(fp)**/
else{
fseek(fp,0L,SEEK_END);
fsize=ftell(fp);
pcfg=(unsigned char *)malloc(fsize+(0+1)*sizeof(unsigned char));
if(pcfg==NULL) {fclose(fp);goto start;}

fseek(fp,0L,SEEK_SET);
fread(pcfg,1,fsize,fp);
cfgmax=fsize/1-1;
fclose(fp);

/* 1 -> 4 : sizes of font cell and window */
if((UDX=read_cfg(1))==-1000) {free(pcfg);goto start;}
if((UDY=read_cfg(2))==-1000) {free(pcfg);goto start;}
if((COLUMN=read_cfg(3))==-1000) {free(pcfg);goto start;}
if((ROW_L=read_cfg(4))==-1000) {free(pcfg);goto start;}

UDX=max(min(UDX,32),4);
UDY=max(min(UDY,64),8);
COLUMN=max(COLUMN,COLUMN_MIN);
ROW_L=max(ROW_L-ROW_L%2,ROW_L_MIN);
if((RIGHT_M=read_cfg(23))==-1000) RIGHT_M=3;
else RIGHT_M=max(min(RIGHT_M,10),1);

while(1){
XRESO=(COLUMN+DI+DI_)*UDX+DX_FRAME*2+UDX*RIGHT_M;
```

```
if(XRESO>XRESO_MAX-UDX*2) COLUMN--;else break;
if(COLUMN<COLUMN_MIN){
COLUMN=COLUMN_MIN;
/*UDX=(XRESO_MAX-DX_FRAME*2)/(COLUMN+DI+DI_+RIGHT_M);*/
XRESO=(COLUMN+DI+DI_)*UDX+DX_FRAME*2+UDX*RIGHT_M;
break;}
}

while(1){
YRESO=(ROW_L+/*2*/3)*UDY+(DY_CAPTION+DY_MENU+DY_TOOLBAR+DY_FRAME*2);

if(YRESO>YRESO_MAX-UDY*2) ROW_L--;else break;
if(ROW_L<ROW_L_MIN){
ROW_L=ROW_L_MIN;
/*UDY=(YRESO_MAX-(DY_CAPTION+DY_MENU+DY_TOOLBAR+DY_FRAME*2))/(ROW_L+3);*/
break;}
}

ROW_L=ROW_L-ROW_L%2;
YRESO=(ROW_L+/*2*/3)*UDY+(DY_CAPTION+DY_MENU+DY_TOOLBAR+DY_FRAME*2);

/* 5 : size of tab */
if((TAB_c=read_cfg(5))==-1000) TAB_c=8;
else TAB_c=max(min(TAB_c,COLUMN-1),1);

/* 6 : size of cursor */
if((CSRDY=read_cfg(6))==-1000) CSRDY=UDY;
else {if(CSRDY==0) CSRDY=UDY;else CSRDY=max(min(CSRDY,UDY),1);}

/* 7 -> 15 : color */
if((WB=read_cfg(7))==-1000) WB=0;
else WB=min(WB,1);

if((ACTIVE=read_cfg(8))==-1000) ACTIVE=13;
else ACTIVE=min(ACTIVE,15);
if((INACTIVE=read_cfg(9))==-1000) INACTIVE=8;
else INACTIVE=min(INACTIVE,15);

if((RTC=read_cfg(10))==-1000) RTC=9;
else RTC=min(RTC,15);

if((RETURN=read_cfg(11))==-1000) RETURN=9;
else RETURN=min(RETURN,15);
if((TABCOLOR=read_cfg(12))==-1000) TABCOLOR=3/*bfset[WB].back*/;  /* 3 : for me */
else TABCOLOR=min(TABCOLOR,15);
if((CC=read_cfg(13))==-1000) CC=9;
```

```
else CC=min(CC,15);

if((CSRCOLOR=read_cfg(14))==-1000) CSRCOLOR=15;
else CSRCOLOR=max(min(CSRCOLOR,15),1);
if((CSRCOLOR_FILER=read_cfg(15))==-1000) CSRCOLOR_FILER=12;
else CSRCOLOR_FILER=max(min(CSRCOLOR_FILER,15),1);

if((AINDENT=read_cfg(16))==-1000) AINDENT=0;
else AINDENT=min(AINDENT,1);

if((fontnum=read_cfg(17))==-1000) fontnum=3;
else fontnum=min(fontnum,3);
if((dh=read_cfg(18))==-1000) dh=0;
/*else dh=min(dh,UDX/2);*/
if((dv=read_cfg(19))/*<0*/==-1000) dv=0;
/*else dv=min(dv,UDY/2);*/

if((l_s_flag=read_cfg(20))==-1000) l_s_flag=0;
else l_s_flag=min(l_s_flag,1);
if((tabspaces=read_cfg(21))==-1000) tabspaces=1;
else tabspaces=min(tabspaces,1);
if((MOVEcsr=read_cfg(22))==-1000) MOVEcsr=1;
else MOVEcsr=min(MOVEcsr,1);
  if(MOVEcsr==0) MOVEcsr=-1;

if((LEFT_m=read_cfg(24))==-1000) LEFT_m=120;
else LEFT_m=min(LEFT_m,XRESO/2);
if((Flag_k=read_cfg(25))==-1000) Flag_k=0;
else Flag_k=min(Flag_k,1);
if((AVMEMDENO=read_cfg(26))==-1000) AVMEMDENO=8;
else AVMEMDENO=max(AVMEMDENO,4);

if((mfsize=read_cfg(27))==-1000) mfsize=/*av_mem/AVMEMDENO*/100;
else mfsize=min(mfsize,/*av_mem/AVMEMDENO*/100);
if((maxfiles=read_cfg(28))==-1000) FMAX=ROW_L+1;
else FMAX=min(max(maxfiles+1,2+1),ROW_L+1);

free(pcfg);
}/**else(fp)**/

arrange_colors();

/*get_av_memory(1,mfsize);*/

ROW_S=(ROW_L+2)/2-2;
/*FMAX=ROW_L+1;*/                       /* maxfiles=FMAX-1 */
```

```
YRESO+=DSHIFT_2*both;
}/** setup **/


int initgraph_(void)
{
if((d=XOpenDisplay(""))==NULL) exit(1);
screen=DefaultScreen(d);

cmap=DefaultColormap(d,screen);
initpalette();

setup();

#if GRP_or_EDT==1
rw=DefaultRootWindow(d);
w=XCreateSimpleWindow(d,rw,/*0*/LEFT_m,0,XRESO,YRESO,0,
                      irgb[bfset[WB].back].pixel,
                      irgb[bfset[WB].back].pixel);

sh.flags=PPosition | PSize;
sh.x=0;sh.y=0;
sh.width=XRESO;sh.height=YRESO;
XSetStandardProperties(d,w,appliname,appliname,None,argv_,argc_,&sh);

XSelectInput(d,w,KeyPressMask | ButtonPressMask | ExposureMask |
                 EnterWindowMask | LeaveWindowMask |
                 StructureNotifyMask | SubstructureNotifyMask | mask);
/*XSelectInput(d,w,KeyPressMask | ExposureMask | mask);*/

/*XMoveWindow(d,w,LEFT_m,0);*/
XMapWindow(d,w);

gcdisplay=XCreateGC(d,w,0,NULL);

depth=DefaultDepth(d,screen);
pmap1=XCreatePixmap(d,w,XRESO,YRESO,depth);  /* text */
pmap3=XCreatePixmap(d,w,XRESO,UDY*3,depth);  /* cursor */

cursor=XCreateFontCursor(d,XC_arrow);
XDefineCursor(d,w,cursor);

initIME();

/*font_fs=XCreateFontSet(d,fs1,&mlist,&mcount,&def);*/  /* -> initIME() */
```

```
XFontsOfFontSet(font_fs,&info,&flist);
asct=(*info)->ascent;
dsct=(*info)->descent;
dv+=asct+dsct+0;

cleardevice_(-1,0,0,0,0);
setcsrcolor((csrcolor=CSRCOLOR));
paint(3,0,2*UDY,XRESO,UDY,14);  /* for icsr_f, jcsr_f in L */
#endif

mallocs();

ftp=0;
fsp=FMAX-1-ftp;
fn=0;                                /* file ID ? */

ROW=ROW_L;DJ=0;

/*XRESO-=DX_FRAME*2;*/
YRESO-=DY_CAPTION+DY_MENU+DY_TOOLBAR+DY_FRAME*2;
YRESO-=DSHIFT_2*both;
YRESO-=UDY-1;

return 0;
}/** initgraph_ **/


void closegraph_(void)
{
int i;

frees();
free(ptmp);

#if GRP_or_EDT==1
XFreeCursor(d,cursor);
XFreeFontSet(d,font_fs);
XFreePixmap(d,pmap1);XFreePixmap(d,pmap3);
XFreeGC(d,gcdisplay);

XFreeColormap(d,cmap);
XDestroyWindow(d,w);/*XFlush(d);*/
XCloseDisplay(d);

if(FF_2/2) fprintf_2(fname_bg);     /* 2, 3 */
#endif
```

```
}/** closegraph_ **/


void initpalette(void)
{
int i;

irgb[0].red=0;irgb[0].green=0;irgb[0].blue=0;

irgb[1].red=0;irgb[1].green=0;irgb[1].blue=127+64;
irgb[2].red=0;irgb[2].green=127+64;irgb[2].blue=0;
irgb[3].red=0;irgb[3].green=127+64;irgb[3].blue=127+64;
irgb[4].red=127+64;irgb[4].green=0;irgb[4].blue=0;
irgb[5].red=127+64;irgb[5].green=0;irgb[5].blue=127+64;
irgb[6].red=127+64;irgb[6].green=127+64;irgb[6].blue=0;
irgb[7].red=127+64;irgb[7].green=127+64;irgb[7].blue=127+64;

irgb[8].red=127;irgb[8].green=127;irgb[8].blue=127;

irgb[9].red=0;irgb[9].green=0;irgb[9].blue=255;
irgb[10].red=0;irgb[10].green=255;irgb[10].blue=0;
irgb[11].red=0;irgb[11].green=255;irgb[11].blue=255;
irgb[12].red=255;irgb[12].green=0;irgb[12].blue=0;
irgb[13].red=255;irgb[13].green=0;irgb[13].blue=255;
irgb[14].red=255;irgb[14].green=255;irgb[14].blue=0;
irgb[15].red=255;irgb[15].green=255;irgb[15].blue=255;

for(i=0;i<VGACOLORS;i++){
irgb[i].red=ff_fc(irgb[i].red*TRANS);
irgb[i].green=ff_fc(irgb[i].green*TRANS);
irgb[i].blue=ff_fc(irgb[i].blue*TRANS);
}

for(i=0;i<VGACOLORS;i++)
XAllocColor(d,cmap,&irgb[i]);
}/** initpalette **/


int ff_fc(double val_d)
{
int val_i,val;

val_i=floor(val_d);
val=(val_d-val_i<0.5)?val_i:(val_i+1);

return val;
```

```
}/** ff_fc **/


void puts_(int i,int j,unsigned char *str)
{
int dx,dy;
int length;

length=strlen(str);

i=i+DI_m;j=j-2;dx=(i+DI)*UDX;dy=(j+DJ)*UDY+DSHIFT_2;    /* large */
paint(0,dx,dy,UDX*(length+2+2),UDY*(1+2+2),7);

i++;j++;dx=(i+DI)*UDX;dy=(j+DJ)*UDY+DSHIFT_2;    /* small */
cleardevice_(0,dx,dy,UDX*(length+2),UDY*(1+2));

i++;j++;
while_puts_show_str(0,ACTIVE,i,j,str);
}/** puts_ */


void while_puts_show_str(char flag,int stccolor,int i,int j,unsigned char *str)
{
char TextOutflag,type;
int dx,dy,dy_;
long k,ksmax;
unsigned char s[1],s_[1];
unsigned char jis[2];

ksmax=strlen(str)-1;

TextOutflag=1;
k=0;

if(flag==0) dy_=DSHIFT_2;else dy_=0;

while(1){
s[0]=str[k];
/*if(s[0]=='\0') break;*/
type=/*gettype(str,k)*/0;

if(type<=2){

if(TextOutflag){
if(s[0]>=0x20 && type==0)
setstccolor(stccolor);
```

```
else if(type==-1)
setstccolor(/*12*/stccolor);
/*else if(s[0]=='\n')
setstccolor(RETURN);*/
/*else if(s[0]==0x09)
setstccolor(TABCOLOR);*/
else
setstccolor(/*CC*/RTC);

dx=(i+DI)*UDX;dy=(j+DJ)*UDY+dy_;

if(s[0]>=0x20 && type==0)
stc(flag,dx,dy,s,1);
/*else if(s[0]=='\n'){
s_[0]=0x0d;
stc(flag,dx,dy,s_,1);
}*/
else if(type==-1){
s_[0]=0x0d;
stc(flag,dx,dy,s_,1);
}
/*else if(s[0]==0x09){
s_[0]=0x0d;
stc(flag,dx,dy,s_,1);
}*/
else{
if(s[0]==0x7f) s[0]=0x00;
s_[0]=cc[s[0]];
stc(flag,dx,dy,s_,1);
}
}/**if(TextOutflag)**/

k++;
i++;
if(/*i==COLUMN*/0) break;
}/**if(type)**/
else if(type==3){
if(TextOutflag){
jis[0]=str[k];
jis[1]=str[k+1];

dx=(i+DI)*UDX;dy=(j+DJ)*UDY+dy_;
setstccolor(stccolor);
stc(flag,dx,dy,jis,2);
}/**if(TextOutflag)**/
```

```
k+=2;
i+=2;
if(/*i>=COLUMN*/0) break;
}/**else if(type)**/
else{
}/**else(type)**/

if(k>ksmax) break;                    /* new break */
}
}/** while_puts_show_str **/


void BitBlt_nomline(void)
{
if(divisionnumber==0)
bitblt(1,0,0,XRESO,YRESO,0,0);
else if(divisionnumber==1)
bitblt(1,0,0,XRESO,(ROW+2)*UDY,0,0);
else
bitblt(1,0,DJ*UDY,XRESO,YRESO-DJ*UDY,0,DJ*UDY);  /* DJ = ROW+2 */

BitBltflag=1;
}/** BitBlt_nomline **/


void BitBlt_full(void)
{
char flag;

if(filerflag==1){
flag=1;
if(topp[jcsr]<topp_floor) {icsr=0;jcsr=jcsr_floor/*+1*/;page_firstk(0);flag=0;}
if(jcsrmax==0) {scroll_up(0);flag=0;} if(jcsr>jcsrmax-1) jcsr=jcsrmax-1;

if(flag){
puts_mline_flag=0;
monitorline(0);
/*if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}*/

/*bitblt(1,0,0,XRESO,YRESO,0,0);*/
if(divisionnumber==0)
bitblt(1,0,0,XRESO,YRESO,0,0);
else if(divisionnumber==1)
bitblt(1,0,0,XRESO,(ROW+2)*UDY,0,0);
else
bitblt(1,0,DJ*UDY,XRESO,YRESO-DJ*UDY,0,DJ*UDY);  /* DJ = ROW+2 */
```

```
extraline(-1);
}
}/**if(filerflag)**/
else{
puts_mline_flag=0;
monitorline(0);
/*if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}*/

if(cut==2) csr_to_1_BL(1);

/*bitblt(1,0,0,XRESO,YRESO,0,0);*/
if(divisionnumber==0)
bitblt(1,0,0,XRESO,YRESO,0,0);
else if(divisionnumber==1)
bitblt(1,0,0,XRESO,(ROW+2)*UDY,0,0);
else
bitblt(1,0,DJ*UDY,XRESO,YRESO-DJ*UDY,0,DJ*UDY);  /* DJ = ROW+2 */

extraline(-1);

if(cut==2 && jcsr_f<ROW && icsr_f!=-1) csr_to_1_BL(0);
}/**else(filerflag)**/

BitBltflag=1;
}/** BitBlt_full **/



void csr_to_1_BL(char flag)
{
int csrcolor_tmp,dy=0;
long k;

if(rependflag==1) return;

/*XSetFunction(d,gcdisplay,GXxor);*/
bitbltflag=1;

if(dialogflag==0){
if(menuflag==1) ;
else if(menuflag==2) ;
else if(filerflag==1) ;
else{
if(flag==1) scan_BL();
/*printf_(jcsr_f);*/
if(jcsr_f<ROW && icsr_f!=-1){
```

```
k=top_icsr(/*firstline+*/jcsr_f,icsr_f);

csrcolor_tmp=(csrcolor>1)?(csrcolor-1):15;
if(00) setcsrcolor(csrcolor_tmp);
XSetFunction(d,gcdisplay,GXxor);

if(flag_2nd==0){
if(gettype_p(k)!=3)
bitblt(-3,0*UDX,/*0*/2*UDY,UDX,CSRDY,                    /* text(single byte) */
         (icsr_f+DI)*UDX,(jcsr_f+DJ)*UDY+(UDY-CSRDY)+dy);
else
bitblt(-3,0*UDX,/*0*/2*UDY,UDX*2,CSRDY,             /* text(double byte,1st) */
         (icsr_f+DI)*UDX,(jcsr_f+DJ)*UDY+(UDY-CSRDY)+dy);
}
else
bitblt(-3,0*UDX,/*0*/2*UDY,UDX*2,CSRDY,             /* text(double byte,2nd) */
         (icsr_f-1+DI)*UDX,(jcsr_f+DJ)*UDY+(UDY-CSRDY)+dy);

XSetFunction(d,gcdisplay,GXcopy);
if(00) setcsrcolor(csrcolor);              /* restore */
}/**if(jcsr_f,icsr_f)**/
}/**else**/
}/**if(dialogflag)**/
else{
}/**else(dialogflag)**/

/*XSetFunction(d,gcdisplay,GXcopy);*/
bitbltflag=0;
}/** csr_to_1_BL **/


  /* while_puts_dline(long kstart,long kend) breaks before line++ */
  /* while_puts_linenumber(long k,long k_) breaks after line++ */
long while_puts_linenumber(long k,long k_)
{
char TextOutflag,tabflag,ccflag,type,ijlineflag,onceflag;
int i,j,dx,dy,itab,icc,TAB;
long line;
unsigned char s[1],s_[1];
unsigned char jis[2];

/*clear_topp();*/

i=0;j=0;
onceflag=0;
tabflag=0;itab=0;                   /* Tab */
```

```
ccflag=0;icc=0;                    /* Control code */
line=0;
/*topp[j]=k;*/

while(1){
s[0]=p[fn][k];
type=gettype_p(k);

if(type<=2){
if(type==1 && onceflag==0) {onceflag=1;TAB=getTAB(i);tabflag=1;}  /* TAB_v */
else if(type==2) ccflag=1;              /* Control code */

if(TextOutflag){
}/**if(TextOutflag)**/

/*if(tabflag==0 && ccflag==0 && k>=kmax[fn]) break;*/    /* Tab, Control code */

if(tabflag==1){                    /* Tab */
itab++;
if(itab==TAB) {onceflag=0;tabflag=0;itab=0;k++;}
}
else if(ccflag==1){                /* Control code */
icc++;
if(icc==2) {ccflag=0;icc=0;k++;}
}
else k++;



if(tabflag==0 && ccflag==0 && k>kmax[fn]) break;    /* new break */



i++;

if(s[0]=='\n'){
ijlineflag=1;
}/**if(s[0])**/
else{
if(tabflag==0 && ccflag==0 && i==COLUMN){
ijlineflag=1;}
else if(tabflag==1 /*&& ccflag==0 */&& i==COLUMN){  /* Tab */
onceflag=0;tabflag=0;itab=0;k++;
ijlineflag=1;}
else if(/*tabflag==0 && ccflag==0 && */i==COLUMN+1){  /* Control code */
ijlineflag=1;}
```

```
else ijlineflag=0;
}/**else(s[0])**/
}/**if(type)**/
else if(type==3){
if(TextOutflag){
}/**if(TextOutflag)**/

/*if(k>=kmax[fn]) break;*/              /* ? */

k+=2;



if(k>kmax[fn]) break;                /* new break */



i+=2;

if(/*i==COLUMN || i==COLUMN+1*/i>=COLUMN)
ijlineflag=1;
else ijlineflag=0;
}/**else if(type)**/
else{
}/**else(type)**/

if(ijlineflag==1){
i=0;j++;
/*if((type<=2 && s[0]=='\n')||(jumpflag==1 && linelength==0)||jumpflag==0)*/
if(linelength_new==0) line++;  /* inside work */
else{
if(LINEMODE==0) line++;
else {if(type<=2 && s[0]=='\n') line++;}
}
/*topp[j]=k;*/
}

/*if(j==ROW+1) break;*/
if(k>=k_) break;
}

/*jcsrmax=min(j,ROW);*/
return /*j*/line;
}/** while_puts_linenumber **/
```

```
void monitorline(char flag)
{
unsigned char fc;
int j,dx,dy,color,val;
long dline_;
double/*long*/ pc;
unsigned char attri[8],cutchar[]="1BL",fchar[][2]={"FS","fs"};

val=function%3;
if(val==0) fc=fchar[l_s_flag][0];
else if(val==1) fc=fchar[l_s_flag][reffunc_REF];
else fc=fchar[l_s_flag][reffunc_REP];

dx=0;j=ROW+1;dy=(j+DJ)*UDY;
cleardevice_(1,dx,dy,XRESO,UDY);

/* %.2f:old */
pc=(double)(topp[/*firstline+*/jcsr]+1)/(kmax[fn]+1);  /* double pc */

if(cut==1)      dline_=topp[/*firstline+*/jcsr]-k_from;
else if(cut==2) dline_=top_icsr(/*firstline+*/jcsr,icsr)-k_from;

if(editflag[fn]>-1) strcpy(attri,"");else strcpy(attri,"RO  ");

if(cut>0){
if(paste>0 && okflag_BL>0)
sprintf(mline," %.2f %d %d C:%c-%ld P:%c-%ld %c:%d %s%s",
        pc,jcsr+1,icsr+1,cutchar[cut],dline_,
        cutchar[paste],dk_cut,fc,function%3,attri,fname);
else
sprintf(mline," %.2f %d %d C:%c-%ld P:%c %c:%d %s%s",
        pc,jcsr+1,icsr+1,cutchar[cut],dline_,
        cutchar[paste],fc,function%3,attri,fname);
}/**if(cut)**/
else{
if(paste>0 && okflag_BL>0)
sprintf(mline," %.2f %d %d C:%c P:%c-%ld %c:%d %s%s",
        pc,jcsr+1,icsr+1,cutchar[cut],
        cutchar[paste],dk_cut,fc,function%3,attri,fname);
else
sprintf(mline," %.2f %d %d C:%c P:%c %c:%d %s%s",
        pc,jcsr+1,icsr+1,cutchar[cut],
        cutchar[paste],fc,function%3,attri,fname);
}/**else(cut)**/

if(mlinecolor==0){
```

```
if(editflag[fn]>-1)
while_puts_show_monitorline(0,ACTIVE,j);
else
while_puts_show_monitorline(1,ACTIVE,j);
}
else
while_puts_show_monitorline(0,INACTIVE,j);

if(flag==0) extraline(2);
else{
bitblt(1,dx,dy,XRESO,UDY,dx,dy);
extraline(1);
}
}/** monitorline **/


void while_puts_show_monitorline(char flag,int stccolor,int j)
{
char TextOutflag,type,plane;
int i,dx,dy,RTC_;
long k;
unsigned char s[1],s_[1];
unsigned char jis[2];

kmax_ml=strlen(mline)-1;

if(flag==0) RTC_=RTC;
else {RTC_=stccolor;stccolor=RTC;}

TextOutflag=1;
i=0;                                /* i=0 */
k=0;

/*if(ROWflag==0) */plane=1;/*else plane=0;*/

while(1){
s[0]=mline[k];
/*if(s[0]=='\0') break;*/
type=gettype_mline(k);

if(type<=2){

if(TextOutflag){
if(s[0]>=0x20 && type==0)
setstccolor(stccolor);
else if(type==-1)
```

```
setstccolor(/*12*/stccolor);
/*else if(s[0]=='\n')
setstccolor(RETURN);*/
/*else if(s[0]==0x09)
setstccolor(TABCOLOR);*/
else
setstccolor(/*CC*//*RTC*/RTC_);

if(ROWflag==0){
dx=(i+DI)*UDX;dy=(j+DJ)*UDY;
}
else{
dx=(i+DI)*UDX;dy=j*UDY;
}

if(s[0]>=0x20 && type==0)
stc(plane,dx,dy,s,1);
/*else if(s[0]=='\n'){
s_[0]=0x0d;
stc(plane,dx,dy,s_,1);
}*/
else if(type==-1){
s_[0]=0x0d;
stc(plane,dx,dy,s_,1);
}
/*else if(s[0]==0x09){
s_[0]=0x0d;
stc(plane,dx,dy,s_,1);
}*/
else{
if(s[0]==0x7f) s[0]=0x00;
s_[0]=cc[s[0]];
stc(plane,dx,dy,s_,1);
}
}/**if(TextOutflag)**/

k++;
i++;
if(i==COLUMN) break;
}/**if(type)**/
else if(type==3){
if(TextOutflag){
jis[0]=mline[k];
jis[1]=mline[k+1];

if(ROWflag==0){
```

```
dx=(i+DI)*UDX;dy=(j+DJ)*UDY;
}
else{
dx=(i+DI)*UDX;dy=j*UDY;
}
setstccolor(stccolor);
stc(plane,dx,dy,jis,2);
}/**if(TextOutflag)**/

k+=2;
i+=2;
if(i>COLUMN-1) break;
}/**else if(type)**/
else{
}/**else(type)**/

if(k>kmax_ml) break;                    /* new break */
}
}/** while_puts_show_monitorline **/


void BL(void)
{
tailcheck();

if(cut==0){
}
else if(cut==1){
csr_row_home();
k_from=topp[/*firstline+*/jcsr]+0;
}
else{
k_from=top_icsr(/*firstline+*/jcsr,icsr);
}

firstk_from=firstk;
icsr_from=icsr;jcsr_from=jcsr;
}/** BL **/


void scan_BL(void)
{
char function_old;

if(k_from>=firstk){
if(jcsrmax<ROW || k_from<topp[ROW]){
```

```
function_old=function;function=2;
jcsr_f=while_puts_dline(firstk,k_from);
function=function_old;
icsr_f=icsr_last;
}
else{
jcsr_f=ROW;
}
}/**if(k_from)**/
else{
jcsr_f=0;
icsr_f=-1;
}/**else(k_from)**/
}/** scan_BL **/


void swap_BL(char flag)
{
char function_old;
long k_;
long ddline;

if(k_to-k_from<0){
k_=k_from;k_from=k_to;k_to=k_;      /* swap */

firstk_from=firstk;                /* new */
icsr_from=icsr;jcsr_from=jcsr;
}
else{                              /* no swap */
if(flag==0){                       /* 'Y' */
function_old=function;function=2;

ddline=get_firstk(k_from,jcsr/*_from*/);  /* modify firstk */
if(ddline<0) jcsr_from=ddline+jcsr/*_from*/;
else jcsr_from=jcsr/*_from*/;

function=function_old;
icsr_from=icsr_last;
firstk_from=firstk;
}
}
}/** swap_BL **/


void YKP(char operation)
{
```

```
char function_old;
char type;
long k;

tailcheck();

if(operation==0){                    /* 'Y' */
if(cut==0){
firstk_from=firstk;
icsr_from=icsr;jcsr_from=jcsr;
k_from=topp[/*firstline+*/jcsr]+0;
k_to=top_icsr(/*firstline+*/jcsr,return_is(/*firstline+*/jcsr));
if(k_to!=kmax[fn]){
type=gettype_p(k_to);
if(type<=2) k_to++;
else if(type==3) k_to+=2;
else ;}
if(k_to-k_from>0){
dk_line=k_to-k_from;
memory(1);
text_to_file(2,0);
paste=0;okflag_1=1;
deletion_dk();}
}/**if(cut)************************/
else if(cut==1){
k_to=topp[/*firstline+*/jcsr]+0;
if(k_to-k_from!=0){
swap_BL(0);
dk_old=dk;dk=k_to-k_from;
if(memory(0)==0){
text_to_file(3,0);
dk_cut=dk;
paste=1;okflag_BL=1;
cut=0;
deletion_dk();}}
}/**else if(cut)*******************/
else{
k_to=top_icsr(/*firstline+*/jcsr,icsr);
if(k_to-k_from!=0){
swap_BL(0);
dk_old=dk;dk=k_to-k_from;
if(memory(0)==0){
text_to_file(4,0);
dk_cut=dk;
paste=2;okflag_BL=1;
cut=0;
```

```
deletion_dk();}}
}/**else(cut)**********************/

cut=0;
}/**if(operation)**/
else if(operation==1){              /* 'K' */
if(cut==0){
k_from=topp[/*firstline+*/jcsr]+0;
k_to=top_icsr(/*firstline+*/jcsr,return_is(/*firstline+*/jcsr));
if(k_to!=kmax[fn]){
type=gettype_p(k_to);
if(type<=2) k_to++;
else if(type==3) k_to+=2;
else ;}
if(k_to-k_from>0){
dk_line=k_to-k_from;
memory(1);
paste=0;okflag_1=1;beep(50);}
}/**if(cut)************************/
else if(cut==1){
k_to=topp[/*firstline+*/jcsr]+0;
if(k_to-k_from!=0){
swap_BL(1);
dk_old=dk;dk=k_to-k_from;
if(memory(0)==0){
dk_cut=dk;
paste=1;okflag_BL=1;
cut=0;
page_firstk/*_from*/(firstk);beep(50);}}
}/**else if(cut)*******************/
else{
k_to=top_icsr(/*firstline+*/jcsr,icsr);
if(k_to-k_from!=0){
swap_BL(1);
dk_old=dk;dk=k_to-k_from;
if(memory(0)==0){
dk_cut=dk;
paste=2;okflag_BL=1;
cut=0;
page_firstk/*_from*/(firstk);beep(50);}}
}/**else(cut)**********************/

cut=0;
}/**else if(operation)**/
else{                               /* 'P' */
if(cut>0) return;
```

```
if(paste==0){
k=topp[/*firstline+*/jcsr]+0;
if(okflag_1) insertion_dk(1,k);
}
else if(paste==1){
k=topp[/*firstline+*/jcsr]+0;
if(okflag_BL) insertion_dk(0,k);
}
else{
k=top_icsr(/*firstline+*/jcsr,icsr);
if(okflag_BL){
lumpflag=1;
insertion_dk(0,k);
lumpflag=0;

if(MOVEcsr==1){
function_old=function;function=2;
jcsr=while_puts_dline(firstk,k+dk);
if(jcsr>ROW-1) {firstk=while_puts_firstk(firstk,jcsr-(ROW-1));jcsr=ROW-1;}
function=function_old;
icsr=icsr_last;}
page_firstk(firstk);
}
}
}/**else(operation)**/
}/** YKP **/


void title(char *str)
{
int i,j,dx,dy,dx_;
int length;

/*strcpy(str,"Error");*/
length=strlen(str);

if(dialogflag){
i=DI_d;j=DJ_d-1;dx=i*UDX;dy=j*UDY;    /* large */
}
else if(menuflag){
i=1+DI_m;j=0;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;    /* large */
}
else{}

setstccolor(0);
```

```
dx_=dx;i=0;
while(1){
dx=dx_+i*UDX;
stc(1,dx,dy,&str[i],1);

i++;
if(i==length) break;
}
}/** title **/


void message(int flag,char nobitbltflag)
{
if(nobitbltflag!=1 && nobitbltflag>-1){
bitblt(1,0,0,XRESO,YRESO,0,0);
}

puts_message(flag);

BitBltflag=0;

messageflag=flag;
use_subroop();                        /* -> usflag = 1, function = 2 */
messageflag=0;

if(nobitbltflag!=2 && nobitbltflag>-1){
bitblt(1,0,0,XRESO,YRESO,0,0);

BitBltflag=1;
}
}/** message **/


void puts_message(int flag)
{
int i,j;
unsigned char buf[ASIZE];

i=0;j=ROW/2;

if(flag==1)
;
else if(flag==2)
;
else if(flag==3)
```

```
puts_(i,j,"Do you close all the files ? (Y/N)");
else if(flag==4)
puts_(i,j,"Do you quit this editorial work ? (Y/N)");
else if(flag==5)
puts_(i,j,"Do you close the file ? (Y/N)");
else if(flag==6)
puts_(i,j,"You can't open a temporary file. (OK)");
else if(flag==7)
puts_(i,j,"Memory space is not left. (OK)");
else if(flag==8)
;
else if(flag==9)
puts_(i,j,"Close any file, and then retry it. (OK)");
else if(flag==10)
puts_(i,j,"Disk space is not left. (OK)");
else if(flag==11)
puts_(i,j,"The current directory moved to your home. (OK)");
else if(flag==12)
;
else if(flag==13)
puts_(i,j,"The file can't be given any changes. (OK)");
else{                                /* flag < 0 */
/*itoa(-flag,buf,10);
strcat(buf," ");
strcat(buf,(unsigned char *)strerror(-flag));*/
strcpy(buf,(unsigned char *)strerror(-flag));
strcat(buf,". (OK)");

puts_(i,j,buf);
}
}/** puts_message **/


void while_puts_show_menu(int j,char flag,unsigned char *buf)
{
char TextOutflag,type;
int i,dx,dy;
long k,kmmax;
unsigned char s[1],s_[1];
unsigned char jis[2];

if(flag==0) kmmax=strlen(fnames[ftable[j-1].fn])-1;
else kmmax=strlen(buf)-1;

TextOutflag=1;
i=4+DI_m;                                /* i=4 */
```

```
k=0;

while(1){
if(flag==0) s[0]=fnames[ftable[j-1].fn][k];
else s[0]=buf[k];
/*if(s[0]=='\0') break;*/
if(flag==0) type=gettype_fnames(j,k);
else type=gettype_buf(k,buf);

if(type<=2){

if(TextOutflag){
if(s[0]>=0x20 && type==0)
setstccolor(bfset[WB].fore);
else if(type==-1)
setstccolor(/*12*/bfset[WB].fore);
/*else if(s[0]=='\n')
setstccolor(RETURN);*/
/*else if(s[0]==0x09)
setstccolor(TABCOLOR);*/
else
setstccolor(/*CC*/RTC);

dx=(i+DI)*UDX;dy=(j+DJ)*UDY;

if(s[0]>=0x20 && type==0)
stc(1,dx,dy,s,1);
/*else if(s[0]=='\n'){
s_[0]=0x0d;
stc(1,dx,dy,s_,1);
}*/
else if(type==-1){
s_[0]=0x0d;
stc(1,dx,dy,s_,1);
}
/*else if(s[0]==0x09){
s_[0]=0x0d;
stc(1,dx,dy,s_,1);
}*/
else{
if(s[0]==0x7f) s[0]=0x00;
s_[0]=cc[s[0]];
stc(1,dx,dy,s_,1);
}
}/**if(TextOutflag)**/
```

```
k++;
i++;
if(i==COLUMN) break;
}/**if(type)**/
else if(type==3){
if(TextOutflag){
if(flag==0){
jis[0]=fnames[ftable[j-1].fn][k];
jis[1]=fnames[ftable[j-1].fn][k+1];
}
else{
jis[0]=buf[k];
jis[1]=buf[k+1];
}

dx=(i+DI)*UDX;dy=(j+DJ)*UDY;
setstccolor(bfset[WB].fore);
stc(1,dx,dy,jis,2);
}/**if(TextOutflag)**/

k+=2;
i+=2;
if(i>=COLUMN) break;
}/**else if(type)**/
else{
}/**else(type)**/

if(k>kmmax) break;                    /* new break */
}
}/** while_puts_show_menu **/


void setstccolor(int color)
{
#if GRP_or_EDT==0
return;
#endif

XSetForeground(d,gcdisplay,irgb[color].pixel);
}/** setstccolor **/


void stc(char flag,int x,int y,unsigned char *str,int size)
{
#if GRP_or_EDT==0
return;
```

```
#endif

if(flag==1)
XmbDrawString(d,pmap1,font_fs,gcdisplay,x,y+XDSDY,str,size);
else
XmbDrawString(d,w,font_fs,gcdisplay,x,y+XDSDY,str,size);
}/** stc **/


void cleardevice_(char flag,int x,int y,int xsize,int ysize)
{
int dx;

XSetForeground(d,gcdisplay,irgb[bfset[WB].back].pixel);

if(cut==0) dx=0;
else dx=UDX;

if(flag==0)
XFillRectangle(d,w,gcdisplay,x,y,xsize,ysize);
else if(flag==-1){
if(divisionnumber==0)
XFillRectangle(d,pmap1,gcdisplay,dx,0,XRESO-dx,YRESO);
else if(divisionnumber==1)
XFillRectangle(d,pmap1,gcdisplay,dx,0,XRESO-dx,(ROW+2)*UDY);
else
XFillRectangle(d,pmap1,gcdisplay,dx,DJ*UDY,XRESO-dx,YRESO-DJ*UDY);
}
else if(flag==1)
XFillRectangle(d,pmap1,gcdisplay,x,y,xsize,ysize);
else
/*XFillRectangle(d,pmap3,gcdisplay,x,y,xsize,ysize);*/
XFillRectangle(d,pmap3,gcdisplay,0,0,xsize,ysize);
}/** cleardevice_ **/


void paint(char flag,int x,int y,int xsize,int ysize,int color)
{
XSetForeground(d,gcdisplay,irgb[color].pixel);

if(flag==0)
XFillRectangle(d,w,gcdisplay,x,y,xsize,ysize);
else if(flag==1)
XFillRectangle(d,pmap1,gcdisplay,x,y,xsize,ysize);
else
XFillRectangle(d,pmap3,gcdisplay,x,y,xsize,ysize);
```

```
}/** paint **/


void setcsrcolor(int color)
{
XSetForeground(d,gcdisplay,irgb[color].pixel);
XFillRectangle(d,pmap3,gcdisplay,0,0,XRESO,UDY);  /* for cursor */
}/** setcsrcolor **/


void restore_3(char flag)
{
if(ftp==0 && refill==0) goto skip;

/*if(restoreflag==0){
if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}
else
extraline(-1);
}
else{
if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}
else
extraline(-1);
}*/

if(dialogflag==0 && menuflag==0 && filerflag==0 && cut==2)
csr_to_1_BL(1);
/*99*/
bitblt(1,0,0,XRESO,YRESO+UDY,0,0);
if(dialogflag==0 && menuflag==0 && filerflag==0 && cut==2 && jcsr_f<ROW && icsr_f!=-1)
csr_to_1_BL(0);
                                    /* csr() */
if(flag==1) csr();
else{
if(dialogflag==0 && menuflag==0 && filerflag==0){
if(cut>0) indicator(1);
else {if(indicationflag) {indicationflag=0;indicator(0);}}
}
else BitBlt_indicator();
}

if(messageflag) puts_message(messageflag);

skip:{}
}/** restore_3 **/
```

```
void indicator(char flag)
{
int jcsr_1,jcsr_2,djcsr;
long k;

if(divisionnumber==0)
cleardevice_(1,0,0,UDX,YRESO);
else if(divisionnumber==1)
cleardevice_(1,0,0,UDX,(ROW+2)*UDY);
else
cleardevice_(1,0,DJ*UDY,UDX,YRESO-DJ*UDY);  /* DJ = ROW+2 */

if(flag==0) goto skip;

if(cut==0) {}
else if(cut==1){
scan_BL();
if(jcsr_f!=jcsr){
jcsr_1=min(jcsr_f,jcsr);jcsr_2=max(jcsr_f,jcsr);
paint(1,0,(jcsr_1+DJ)*UDY,UDX,(jcsr_2-jcsr_1)*UDY,ACTIVE);

indicationflag=1;
}
}/**else if(cut)**/
else{
scan_BL();
if(jcsr_f!=jcsr || icsr_f!=icsr){
jcsr_1=min(jcsr_f,jcsr);jcsr_2=max(jcsr_f,jcsr);
if(jcsr_f<ROW) djcsr=jcsr_2-jcsr_1+1;else djcsr=jcsr_2-jcsr_1;
paint(1,0,(jcsr_1+DJ)*UDY,UDX,djcsr*UDY,ACTIVE);

indicationflag=1;
}
}/**else(cut)**/

skip:

BitBlt_indicator();
}/** indicator **/


void BitBlt_indicator(void)
{
int dy=DSHIFT_2;
```

```
if(divisionnumber==0)
XCopyArea(d,pmap1,w,gcdisplay,0,0,UDX,YRESO,0,0+dy);
else if(divisionnumber==1)
XCopyArea(d,pmap1,w,gcdisplay,0,0,UDX,(ROW+2)*UDY,0,0+dy);
else
                                                                    /* DJ = ROW+2 */
XCopyArea(d,pmap1,w,gcdisplay,0,DJ*UDY,UDX,YRESO-DJ*UDY,0,DJ*UDY+dy);
}/** BitBlt_indicator **/


void bitblt(char flag,int x,int y,int xsize,int ysize,int x_,int y_)
{
int dy_=DSHIFT_2;

y_+=dy_;

if(bitbltflag==0){
/*if(flag==0) {}
else */if(flag==1)
XCopyArea(d,pmap1,w,gcdisplay,x,y,xsize,ysize,x_,y_);
else if(flag==3)                    /* flag = 3 */ /* for csr() */
XCopyArea(d,pmap1,w,gcdisplay,/*x*/x_,/*y*/y_-dy_,xsize,ysize,x_,y_);
}/**if(bitbltflag)**/
else{
/*if(flag==0) {}
else */if(flag==1)
XCopyArea(d,pmap1,w,gcdisplay,x,y,xsize,ysize,x_,y_);
else if(flag==-3)           /* flag = -3 */ /* for csr_to_1(), for csr_to_1_BL() */
XCopyArea(d,pmap3,pmap1,gcdisplay,x,y,xsize,ysize,x_,y_-dy_);
}/**else(bitbltflag)**/

XFlush(d);
}/** bitblt **/


void extraline(char flag)
{
int i,j,dx,dy;
/*static */unsigned char IorO[]="IO";

if(no_extraline) return;

if(cqflag==0 && puts_mline_flag>0) return;
puts_mline_flag=0;

dx=0;j=ROW_L+2;dy=j*UDY;
```

```
if(flag>-1) cleardevice_(1,dx,dy,XRESO,UDY);

if(flag>0/* && refflag==0*/ && (ftp>0 || filerflag==1)){
if(refflag==0){
if(filerflag==1 || deletedflag==1){
if(strlen(ref_t)==0)
sprintf(mline,"  %c  No string",IorO[insorover]);
else
sprintf(mline,"  %c  String = \"%s\"",IorO[insorover],ref_t);
}/**if(filerflag)**/
else{
if(strlen(ref_t)==0)
sprintf(mline,"  %c  N:%d  No string",IorO[insorover],ftp);
else
sprintf(mline,"  %c  N:%d  String = \"%s\"",IorO[insorover],ftp,ref_t);
}/**else(filerflag)**/
}/**if(refflag)**/
else{
sprintf(mline,"  %c",IorO[insorover]);
}/**else(refflag)**/

ROWflag=1;
while_puts_show_monitorline(0,ACTIVE,j);
ROWflag=0;
}

/*if(function!=2 && flag>-1) putstrings();*/

if(flag!=2) bitblt(1,dx,dy,XRESO,UDY,dx,dy);
}/** extraline **/



void BitBlt_menu(void)
{
int i,j,dx,dy;

i=0+DI_m;j=0;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;    /* large */

if(menuflag==1)
bitblt(1,dx,dy,UDX*(sizeoffname+3+5+2),UDY*(ftp+2),dx,dy);
else
bitblt(1,dx,dy,UDX*(12+4+2),UDY*(3+2),dx,dy);

/*if(menuflag==1 && cqflag>0) {extraline(1);cqflag=0;}*/  /* no problem ? */

BitBltflag_=1;
```

```
}/** BitBlt_menu **/


void before_mainroop_menu_REP(char *str)
{
char menuflag_old;
int i,j,dx,dy;

cleardevice_(-1,0,0,0,0);
while_puts_show_(1,firstk);         /* erase the dialog window */
monitorline(0);

menuflag_old=menuflag;menuflag=0;

if(dialogflag==0 && menuflag==0 && filerflag==0){
if(cut>0) indicator(1);
else {if(indicationflag) {indicationflag=0;indicator(0);}}
}
else BitBlt_indicator();
if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

menuflag=menuflag_old;


i=0+DI_m;j=0;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;    /* large */
paint(1,dx,dy,UDX*(12+4+2),UDY*(3+2),7);

title(str);                          /* title */

i++;j++;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;    /* small */
cleardevice_(1,dx,dy,UDX*(12+4),UDY*3);

i+=2;
while_puts_show_str(1,bfset[WB].fore,i,j,"All the text");
j++;
while_puts_show_str(1,bfset[WB].fore,i,j,"Forward");
j++;
while_puts_show_str(1,bfset[WB].fore,i,j,"Backward");

/*BitBlt_menu();*/
/*BitBlt_full();*/
/*BitBlt_nomline();*/                  /* erase the dialog window */
bitblt(1,0,0,XRESO,YRESO,0,0);
extraline(-1);

icsr=2;jcsr=1;
```

```
csr();
}/** before_mainroop_menu_REP **/


void before_mainroop_menu(char *str)
{
int i,j,dx,dy;
unsigned char s[1];
unsigned char buf[(CD+1)-8+1];  /* +1 : because of strlen() */

if(divisionnumber==1)
fn_1st=fn;
else if(divisionnumber==2)
fn_2nd=fn;
else{}

sizeoffname=min(max(sizeoffname,20),sizeoffname_max);

i=0+DI_m;j=0;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;    /* large */
paint(1,dx,dy,UDX*(sizeoffname+3+5+2),UDY*(ftp+2),7);

title(str);                          /* title */

i++;j++;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;    /* small */
cleardevice_(1,dx,dy,UDX*(sizeoffname+3+5),UDY*ftp);

j=1;
while(1){
/*setstccolor(bfset[WB].fore);*/

if(strlen(fnames[ftable[j-1].fn])<=sizeoffname_max)
while_puts_show_menu(j,0,"");         /* fnames[] */
else{
i=4+DI_m;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;
strcpy(buf,fnames_shortened(j));
strcat(buf,"<");

/*stc(1,dx,dy,buf,strlen(buf));*/
while_puts_show_menu(j,1,buf);         /* buf */
}

if(editflag[ftable[j-1].fn]==1){
setstccolor(bfset[WB].fore);
i=2+DI_m;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;s[0]='*';  /* if edited */
stc(1,dx,dy,s,1);
}
```

```
else if(editflag[ftable[j-1].fn]<=-1){
setstccolor(RTC);
i=2+DI_m;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;s[0]='r';  /* read-only */
stc(1,dx,dy,s,1);
}

if(divisionnumber>0){
if(ftable[j-1].fn==fn_1st){
if(divisionnumber==1)
setstccolor(ACTIVE);
else
setstccolor(INACTIVE);
i=1+DI_m+sizeoffname+3+5-2;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;s[0]='1';
stc(1,dx,dy,s,1);
}
else if(ftable[j-1].fn==fn_2nd){
if(divisionnumber==2)
setstccolor(ACTIVE);
else
setstccolor(INACTIVE);
i=1+DI_m+sizeoffname+3+5-2;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;s[0]='2';
stc(1,dx,dy,s,1);
}
}

if(divideflag_==2 && j==jcsr_select){
setstccolor(ACTIVE);
i=1+DI_m+sizeoffname+3+5-4;dx=(i+DI)*UDX;dy=(j+DJ)*UDY;s[0]='1';
stc(1,dx,dy,s,1);
}

j++;
if(j==ftp+1) break;
}

BitBlt_menu();

icsr=2;jcsr=ftp-1+1;
csr();
}/** before_mainroop_menu **/


unsigned char *fnames_shortened(int j)
{
char type;
int length;
```

```
long k,k_first;
static unsigned char buf[(CD+1)-8/*+1*/];

length=strlen(fnames[ftable[j-1].fn]);

k=0;k_first=0;
while(1){
if(k>length-1) break;
if(fnames[ftable[j-1].fn][k]=='/') k_first=k+1;

type=gettype_fnames(j,k);
if(type<=2) k+=1;
else if(type==3) k+=2;
else{}
}

k=k_first;
while(1){
if(type<=2) {if(k-k_first+1>sizeoffname_max-1) break;}
else {if(k-k_first+1>sizeoffname_max-2) break;}
if(k>length-1) break;

type=gettype_fnames(j,k);
if(type<=2) k+=1;
else if(type==3) k+=2;
else{}
}

strncpy(buf,&fnames[ftable[j-1].fn][k_first],k-k_first);
buf[k-k_first]='\0';

return buf;                          /* static unsigned char *buf */
}/** fnames_shortened **/


void before_mainroop_(char *str)
{
int length;

length=strlen(p_dialog);
if(length<ASIZEM){
p_dialog[length]=0x1a;
p_dialog[length+1]='\0';
}
else{}                               /* impossible */
```

```
strcpy(p_restore,p_dialog);

kmax_dialog=0;
p_dialog[0]=0x1a;
p_dialog[1]='\0';

/*within_linemax_dialog();*/
tailcheck_dialog();
/*firstk_dialog=max(min(firstk_dialog,kmax_dialog),0);*/   /* protection */
}/** before_mainroop_ **/


void before_mainroop(char *str)
{
int i,j,dx,dy;
int length;

icsr=0;jcsr=0;

i=DI_d-1;j=DJ_d-1;dx=i*UDX;dy=j*UDY;    /* large */
paint(1,dx,dy,UDX*((CD+1)+2),UDY*(1+2),7);

title(str);                          /* title */

length=strlen(p_dialog);
if(length<ASIZEM){
p_dialog[length]=0x1a;
p_dialog[length+1]='\0';
}
else{}                              /* impossible */

strcpy(p_restore,p_dialog);

if(!driveflag){
if(noclearflag==0) clear_dialog(0);
else{                                /* in dlgproc_SAVE() */
kmax_dialog=strlen(p_dialog)-1;
text_end_dialog();}

csr();
}
}/** before_mainroop **/


void after_mainroop_menu(void)
{
```

```
refill=1;
}/** after_mainroop_menu **/


void after_mainroop(void)
{
refill=1;
}/** after_mainroop **/


void mnuproc_REP(char *str)
{
int icsr_old,jcsr_old;

icsr_old=icsr;jcsr_old=jcsr;

before_mainroop_menu_REP(str);
mainroop();                          /* p_dialog */
after_mainroop_menu();

if(menuflag==3){
menuflag=0;
icsr=icsr_old;jcsr=jcsr_old;
/*BitBlt_full();csr();*/
refill=0;
}

else if(menuflag==2){
/*menuflag=0;*/                      /* <ble204> */
jcsr_select=jcsr;
/*BitBlt_full();csr();*/
}
}/** mnuproc_REP **/


void mnuproc_MULTIFILE(char *str)
{
int icsr_old,jcsr_old,DJ_old;

icsr_old=icsr;jcsr_old=jcsr;

menuflag=1;
DJ_old=DJ;DJ=0;
imm_pause();

before_mainroop_menu(str);
```

```
mainroop();                          /* p_dialog */
after_mainroop_menu();

imm_restart();

if(menuflag==3){
menuflag=0;
DJ=DJ_old;
icsr=icsr_old;jcsr=jcsr_old;
/*BitBlt_full();csr();*/
refill=0;
}

else if(menuflag==2){
menuflag=0;
DJ=DJ_old;
jcsr_select=jcsr;
icsr=icsr_old;jcsr=jcsr_old;
/*BitBlt_full();csr();*/
}
}/** mnuproc_MULTIFILE **/

/************************** menu functions -> ***************************/

void csr_column_home_menu(void)
{
jcsr=1;
}/** csr_column_home_menu **/


void csr_column_end_menu(void)
{
if(menuflag==1){
jcsr=ftp;
}
else{
jcsr=3;
}
}/** csr_column_end_menu **/


void csr_down_menu(void)
{
jcsr++;
if(menuflag==1){
if(jcsr>ftp) {jcsr=ftp;/*scroll_down(0);*/}
```

```
}
else{
if(jcsr>3) {jcsr=3;/*scroll_down(0);*/}
}
}/** csr_down_menu **/


void csr_up_menu(void)
{
jcsr--;
if(jcsr<1) {jcsr=1;/*scroll_up(0);*/}
}/** csr_up_menu **/

/************************** <- menu functions ***************************/


void execute(unsigned char *exefile)
{
/*unsigned char *and=" &";*/

strcat(exefile," &");

if(systemflag) system(exefile);
else{
/*CreateProcess(NULL,exefile,NULL,NULL,0,0,NULL,NULL,&sui,&pi);*/
system(exefile);
/*spawnlp(P_NOWAIT,exefile,exefile,NULL);*/
}
}/** execute **/


void move_and_paste(void)
{
char flag_;
long k;

if(divisionnumber==0) return;

if(divisionnumber==2){
if(editflag[fn_1st]<=-1) return;

string_visible();
switch_division(0);
}
else{
if(editflag[fn_2nd]<=-1) return;
```

```
string_visible();
switch_division(1);
}

if(strlen(array)!=0){
tailcheck();

/*while_puts_show_(0,firstk);*/
k=top_icsr(/*firstline+*/jcsr,icsr);
flag_=pdata_increase(k,&array[0],strlen(array));
/*printf_(k);use_subroop();*/

if(flag_==0) {if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;}
}
else{
/*beep(500);*/
}

page_firstk(firstk);
}/** move_and_paste **/


void use_selector(char flag)
{
char flag_;
char dialogflag_old,function_old,charflag_old,cut_old;
long k_from_old,k;

/*refflag=1;*/

dialogflag_old=dialogflag;dialogflag=0;

if(flag>=3)
write_3vals(ftp-1);

function_old=function;function=0;
charflag_old=charflag;
cut_old=cut;cut=0;
k_from_old=k_from;
/*extraline(0);*/

if(flag==0) ref();
else if(flag==1)  filename();
else if(flag==2)  jump();
else if(flag==3){
```

```
program();                          /* not in dialog */
if(strlen(array)!=0) /*system(array);*//*WinExec(array,1);*/execute(array);
}
else if(flag==4) edit_cfg();        /* not in dialog */
else if(flag==5) /*edit_tmpcfg()*/;     /* not in dialog */
else if(flag==6) copy_string();          /* not in dialog */

function=function_old;
charflag=charflag_old;
cut=cut_old;
k_from=k_from_old;
if(flag==1) {if(strlen(array)==0) ;else extraline(1);}
else extraline(1);

if(flag>=3){
read_3vals(ftp-1);
if(/*flag==6*/0){
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);

if(strlen(array)!=0){
tailcheck();

while_puts_show_(0,firstk);
k=top_icsr(/*firstline+*/jcsr,icsr);
flag_=pdata_increase(k,&array[0],strlen(array));
/*printf_(k);use_subroop();*/

if(flag_==0) {if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;}
}

page_firstk(firstk);
}
else show_top(0);
}

dialogflag=dialogflag_old;
}/** use_selector **/


void use_deleted(char flag)
{
char /*dialogflag_old,*/function_old,charflag_old,reffunc_REF_old;
unsigned char direction_old_old;
int nest_old;
unsigned char ref_s_old[ASIZE],ref_t_old[ASIZE];
```

```
/*dialogflag_old=dialogflag;dialogflag=0;*/

write_3vals(ftp-1);

function_old=function;function=0;
charflag_old=charflag;
reffunc_REF_old=reffunc_REF;
direction_old_old=direction_old;
nest_old=nest;nest=0;
strcpy(ref_s_old,ref_s);
strcpy(ref_t_old,ref_t);

if(flag==0) deleted();
else /*gather()*/;

function=function_old;
charflag=charflag_old;
reffunc_REF=reffunc_REF_old;
direction_old=direction_old_old;
nest=nest_old;
if(0){
strcpy(ref_s,ref_s_old);
strcpy(ref_t,ref_t_old);
}
/*extraline(1);*/

read_3vals(ftp-1);
show_top(0);

/*dialogflag=dialogflag_old;*/
}/** use_deleted **/


void use_filer(void)
{
char dialogflag_old,function_old,charflag_old,cut_old,paste_old,okflag_1_old,
     reffunc_REF_old,l_s_flag_old,newopen_old;
unsigned char direction_old_old;
int nest_old;
long k_from_old;
long dk_line_old;
unsigned char ref_s_old[ASIZE],ref_t_old[ASIZE];

if(ROW<4) return;
```

```
filerflag=1;

dialogflag_old=dialogflag;dialogflag=0;

function_old=function;function=0;
charflag_old=charflag;
reffunc_REF_old=reffunc_REF;
direction_old_old=direction_old;
l_s_flag_old=l_s_flag;l_s_flag=1;
cut_old=cut;cut=0;
k_from_old=k_from;
paste_old=paste;
okflag_1_old=okflag_1;dk_line_old=dk_line;strncpy(buf_line,ptmp_line,dk_line_old);
nest_old=nest;nest=0;
strcpy(ref_s_old,ref_s);
strcpy(ref_t_old,ref_t);
strcpy(ref_s,"");
strcpy(ref_t,"");
/*extraline(1);*/
imm_pause();

filer();

function=function_old;
charflag=charflag_old;
reffunc_REF=reffunc_REF_old;
direction_old=direction_old_old;
l_s_flag=l_s_flag_old;
cut=cut_old;
k_from=k_from_old;
paste=paste_old;
okflag_1=okflag_1_old;dk_line=dk_line_old;strncpy(ptmp_line,buf_line,dk_line_old);
nest=nest_old;
strcpy(ref_s,ref_s_old);
strcpy(ref_t,ref_t_old);
imm_restart_filer();

nest_free_flag=0;filerskip=0;

if(refill_old>-2){                  /* Esc, Shift+Esc */
if(ftp>0){
read_3vals(ftp-1);
/*show_top(1);*/
restore_page_and_oldcsr();
extraline(1);
}
```

```
else{
cleardevice_(-1,0,0,0,0);
BitBlt_nomline();
extraline(0);
}
}
else{                                /* Enter, Shift+Enter */
filerflag=1;                         /* ! */
extraline(1);
filerflag=0;
}

dialogflag=dialogflag_old;
}/** use_filer **/


void restore_page_and_oldcsr(void)
{
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
cleardevice_(-1,0,0,0,0);
while_puts_show_(1,firstk);          /* erase the dialog window */
monitorline(0);

if(dialogflag==0 && menuflag==0 && filerflag==0){
if(cut>0) indicator(1);
else {if(indicationflag) {indicationflag=0;indicator(0);}}
}
else BitBlt_indicator();
if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

bitblt(1,0,0,XRESO,YRESO,0,0);
}/** restore_page_and_oldcsr **/


void dlgproc_OPEN(char flag_rn)
{
int icsr_old,jcsr_old;
long firstk_old;
/*unsigned char fname_old[ASIZE];*/
unsigned char oldstring[ASIZE]="";

if(ftp>0) {if(cut==2) csr_to_1_BL(1);csr_to_1();}  /* write csr to hdctmp1 */

/*fname[0]='\0';*/
firstk_old=firstk;
```

```
icsr_old=icsr;jcsr_old=jcsr;

while(1){
dialogflag=1;

strcpy(p_dialog,fname);             /* to p_dialog */
if(flag_rn==0) before_mainroop("Open");
else if(flag_rn==1) before_mainroop("Open (r)");
else before_mainroop("Open (n)");
mainroop();                         /* p_dialog */
after_mainroop();
strcpy(array,p_dialog);             /* to array */

if(dialogflag==3) break;

/*strcpy(fname_old,array);*/
if(use_selector_flag==1) use_selector(1);
if((arraycheckflag=arraycheck())>1) use_filer();

if(arraycheck()>0){
/*fname[0]='\0';*/strcpy(fname,/*fname_old*/oldstring);
if(strlen(array)>0) puts_mline(0,"The directory is bad.");}
else {strcpy(fname,array);break;}
}/**while(1)**/

if(dialogflag==3){
dialogflag=0;
strcpy(fname,oldstring);
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
/*BitBlt_full();csr();*/
refill=0;
}

else if(dialogflag==2){
dialogflag=0;
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
/*BitBlt_full();csr();*/
}
}/** dlgproc_OPEN **/


void dlgproc_JUMP(void)
{
int icsr_old,jcsr_old;
```

```
long firstk_old;
long linefrom1;
char oldstring[11];

if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

strcpy(oldstring,linestring);
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;

start:

dialogflag=1;

strcpy(p_dialog,linestring);        /* to p_dialog */
if(u_s_flag){
u_s_flag=0;dialogflag=2;use_selector_flag=1;
before_mainroop_("Jump");
trim_dialog();
}
else{
before_mainroop("Jump");
mainroop();                         /* p_dialog */
}
after_mainroop();
if(strlen(p_dialog)<11) strcpy(linestring,p_dialog);        /* to array */
else strcpy(linestring,"1");

if(dialogflag==3){
dialogflag=0;
strcpy(linestring,oldstring);
icsr=icsr_old;jcsr=jcsr_old;
/*BitBlt_full();*/page_firstk(firstk);
}

else if(dialogflag==2){                 /* job */
dialogflag=0;

/*strcpy(linestring_old,linestring);*/

if(strlen(linestring)==0 || linestringcheck()==1 || use_selector_flag==1){
use_selector(2);
if(strlen(array)<11) strcpy(linestring,array);
else strcpy(linestring,"1");

fn=ftable[ftp-1].fn;
```

```
strcpy(fname,fnames[fn]);

if(strlen(linestring)==0 || linestringcheck()==1){
firstk=firstk_old;                  /* read_3vals() */
icsr=icsr_old;jcsr=jcsr_old;
/*page_firstk(firstk);
csr();*/
restore_page_and_oldcsr();
strcpy(linestring,/*linestring_old*/oldstring);goto start;
}
}

icsr=0;jcsr=0;
linelength_new=1;
linefrom1=atol(linestring);if(linefrom1<1) linefrom1=1;
firstk=while_puts_firstk(0,linefrom1-1);
if(linefrom1-1!=line_end) get_firstk(kmax[fn],0);
linelength_new=0;
page_firstk(firstk);
/*csr();*/
}
}/** dlgproc_JUMP **/


int linestringcheck(void)
{
int i,length;

length=strlen(linestring);

if(length==1){
if(linestring[0]<0x30 || linestring[0]>0x39) return 1;
else return 0;
}

i=0;
while(1){
if(linestring[i]<0x30 || linestring[i]>0x39) return 1;

i++;
if(i==length) return 0;
}
}/** linestringcheck **/


void dlgproc_SAVE_(void)
```

```
{
int fn_old;
unsigned char oldstring[ASIZE];

fn_old=fn;
strcpy(fname,fnames[fn]);
strcpy(oldstring,fname);

page_firstk(firstk);
csr();
if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

while(1){
start:

noclearflag=1;
dialogflag=1;

strcpy(p_dialog,fname);             /* to p_dialog */
before_mainroop("Save");
mainroop();                         /* p_dialog */
after_mainroop();
strcpy(array,p_dialog);             /* to array */

if(dialogflag==3) break;

/*strcpy(fname_old,array);*/
if(use_selector_flag==1) use_selector(1);
if((arraycheckflag=arraycheck())>1) use_filer();

if(arraycheck()>0){
/*fname[0]='\0';*/strcpy(fname,/*fname_old*/oldstring);
if(strlen(array)>0) puts_mline(0,"The directory is bad.");}
else {strcpy(fname,array);break;}
}/**while(1)**/

noclearflag=0;

if(dialogflag==3){
dialogflag=0;
}

else if(dialogflag==2){                  /* job */
dialogflag=0;
fn=fn_old;
/*strcpy(fnames[fn],fname);*/
```

```
if(fsave(1,0)==1){
strcpy(fname,/*fname_old*/oldstring);
/*message(2,1);*/puts_mline(0,"Reinput a filename.");goto start;}
strcpy(fnames[fn],fname);
sizeoffname=max(strlen(fname),sizeoffname);
}
}/** dlgproc_SAVE_ **/


void dlgproc_REN(void)
{
int icsr_old,jcsr_old;
long firstk_old;
unsigned char oldstring[ASIZE];

if(editflag[fn]<=-1) {message(13,1);csr();return;}

if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

/*fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);*/
strcpy(oldstring,fname);
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;

while(1){
start:

noclearflag=1;
dialogflag=1;

strcpy(p_dialog,fname);             /* to p_dialog */
before_mainroop("Rename");
mainroop();                         /* p_dialog */
after_mainroop();
strcpy(array,p_dialog);             /* to array */

if(dialogflag==3) break;

/*strcpy(fname_old,array);*/
if(use_selector_flag==1) use_selector(1);
if((arraycheckflag=arraycheck())>1) use_filer();

if(arraycheck()>0){
/*fname[0]='\0';*/strcpy(fname,/*fname_old*/oldstring);
if(strlen(array)>0) puts_mline(0,"The directory is bad.");}
```

```
else {strcpy(fname,array);break;}
}/**while(1)**/

noclearflag=0;

if(dialogflag==3){
dialogflag=0;
strcpy(fname,oldstring);
fn=ftable[ftp-1].fn;
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
page_firstk(firstk);
/*BitBlt_full();csr();*/
}

else if(dialogflag==2){                  /* job */
dialogflag=0;
fn=ftable[ftp-1].fn;
strcpy(fnames[fn],fname);
/*if(fsave(1,0)==1){
fname[0]='\0';message(2,1);goto start;}*/
sizeoffname=max(strlen(fname),sizeoffname);
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
page_firstk(firstk);
/*BitBlt_full();csr();*/
editflag[fn]=1;
}
}/** dlgproc_REN **/


void dlgproc_SAVE(char flag_append)
{
int icsr_old,jcsr_old;
long firstk_old;
unsigned char oldstring[ASIZE];

if(editflag[fn]<=-1) {message(13,1);csr();return;}

if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

/*fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);*/
strcpy(oldstring,fname);
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;
```

```
while(1){
start:

noclearflag=1;
dialogflag=1;

strcpy(p_dialog,fname);             /* to p_dialog */
if(flag_append==0) before_mainroop("Save");
else before_mainroop("Save (a)");
mainroop();                         /* p_dialog */
after_mainroop();
strcpy(array,p_dialog);             /* to array */

if(dialogflag==3) break;

/*strcpy(fname_old,array);*/
if(use_selector_flag==1) use_selector(1);
if((arraycheckflag=arraycheck())>1) use_filer();

/*printf_(arraycheckflag);*/

if(arraycheck()>0){
/*fname[0]='\0';*/strcpy(fname,/*fname_old*/oldstring);
if(strlen(array)>0) puts_mline(0,"The directory is bad.");}
else {strcpy(fname,array);break;}
}/**while(1)**/

noclearflag=0;

if(dialogflag==3){
dialogflag=0;
strcpy(fname,oldstring);
fn=ftable[ftp-1].fn;
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
page_firstk(firstk);
/*BitBlt_full();csr();*/
}

else if(dialogflag==2){                 /* job */
dialogflag=0;
fn=ftable[ftp-1].fn;
/*strcpy(fnames[fn],fname);*/
if(fsave(1,flag_append)==1){
strcpy(fname,/*fname_old*/oldstring);
```

```
/*message(2,1);*/puts_mline(0,"Reinput a filename.");goto start;}
strcpy(fnames[fn],fname);
sizeoffname=max(strlen(fname),sizeoffname);
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
page_firstk(firstk);
/*BitBlt_full();csr();*/
}
}/** dlgproc_SAVE **/


void dlgproc_INS(void)
{
int icsr_old,jcsr_old;
long firstk_old;
unsigned char oldstring[ASIZE];

if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

tailcheck();
strcpy(oldstring,ins);
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;

while(1){
start:

dialogflag=1;

strcpy(p_dialog,ins);               /* to p_dialog */
before_mainroop("Insert");
mainroop();                         /* p_dialog */
after_mainroop();
strcpy(array,p_dialog);             /* to array */

if(dialogflag==3) break;

/*strcpy(ins_old,array);*/
if(use_selector_flag==1) use_selector(1);
if((arraycheckflag=arraycheck())>1) use_filer();

if(arraycheck()>0){
/*ins[0]='\0';*/strcpy(ins,/*ins_old*/oldstring);
if(strlen(array)>0) puts_mline(0,"The directory is bad.");}
else {strcpy(ins,array);break;}
}/**while(1)**/
```

```
if(dialogflag==3){
dialogflag=0;
strcpy(ins,oldstring);
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
page_firstk(firstk);
/*BitBlt_full();csr();*/
}

else if(dialogflag==2){                  /* job */
dialogflag=0;
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);

firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
while_puts_show_(0,firstk);

if(file_to_text()==1){              /* after icsr and jcsr */
strcpy(ins,/*ins_old*/oldstring);
/*message(1,1);*/puts_mline(0,"Reinput a filename.");goto start;}
page_firstk(firstk);
/*BitBlt_full();csr();*/
}
}/** dlgproc_INS **/


void FILE_jump(char flag)
{
int j,dx,dy,val;
long kend,linefrom1,linemax;
unsigned char buf[11],home[ASIZE];

/*tailcheck();*/
if(jcsr>jcsrmax) jcsr=jcsrmax;

linelength_new=1;
kend=topp[/*firstline+*/jcsr];
linefrom1=while_puts_dline(0,kend)+1;
kend=kmax[fn];
linemax=while_puts_dline(0,kend)+1;
linelength_new=0;
```

```
if(flag==1){
strcpy(home,home_global);
strcat(home,"zzz.jump");
/*ltoa(linefrom1,buf,10);*/gcvt(linefrom1,11,buf);

fpf=fopen(home,"ab");
fwrite(buf,1,strlen(buf),fpf);
fwrite(&two[4][0],1,1,fpf);
fclose(fpf);

beep(50);
}

dx=0;j=ROW_L+2;dy=j*UDY;
cleardevice_(1,dx,dy,XRESO,UDY);
val=(int)(((double)linefrom1/linemax)*100);
sprintf(mline,"  Line Info:%ld/%ld=%01d.%02d",
        linefrom1,linemax,val/100,val%100);

ROWflag=1;
while_puts_show_monitorline(0,ACTIVE,j);
ROWflag=0;
bitblt(1,dx,dy,XRESO,UDY,dx,dy);

BitBltflag=2/*0*/;
/*no_extraline=1;*/
}/** FILE_jump **/



void puts_mline(char flag,char *str)
{
int j,dx,dy;

dx=0;j=ROW_L+2;dy=j*UDY;
cleardevice_(1,dx,dy,XRESO,UDY);

if(flag==0)
sprintf(mline,"  %s",
        str);
else if(flag==1)
sprintf(mline,"  %d %s",
        repcount,str);
else
sprintf(mline,"  R:%d \"%s\" -> \"%s\" %s",
        repcount,ref_t,rep_t_,str);
```

```
ROWflag=1;
while_puts_show_monitorline(0,ACTIVE,j);
ROWflag=0;
bitblt(1,dx,dy,XRESO,UDY,dx,dy);

BitBltflag=2;
if(function!=2) puts_mline_flag=1;
}/** puts_mline **/


void prompt_cq(char flag)
{
int j,dx,dy;

/*if(menuflag==1 && cqflag>0)
{dx=0;j=ROW_L+1;dy=(j+0)*UDY;}
else*/
/*{*/dx=0;j=ROW_L+2;dy=j*UDY;/*}*/
cleardevice_(1,dx,dy,XRESO,UDY);

if(flag==0)
strcpy(mline,"  Cntrol code : @A...Z[\\]^_");
else if(flag==1){
if(cut>0)
strcpy(mline,"  Esc(F12) : S,A");
else
strcpy(mline,"  Esc(F12) : O,C,S(G),E(T),Q(U),W(Y),X(V),R,I,A,N,M");
}
else if(flag==2)
strcpy(mline,"  ^Q");
else
strcpy(mline,"  ~^Q");

ROWflag=1;
while_puts_show_monitorline(0,ACTIVE,j);
ROWflag=0;
bitblt(1,dx,dy,XRESO,UDY,dx,dy);

BitBltflag=2;
/*puts_mline_flag=1;*/
imm_pause();
}/** prompt_cq **/


void FILE_filename(void)
{
```

```
int dm;
int j,dx,dy;
long member,member_,member_Return,member_Start;
unsigned char buf[ASIZE],buf_[ASIZE],home[ASIZE];

member=topp[/*firstline+*/jcsr]+0;

while(1){
if(member==kmax[fn]) goto end;
if(p[fn][member]=='\n') break;

member++;
}

member_Return=member;
if(member>0) member--;
if(p[fn][member]=='\n') goto end;
member_=member;

while(1){
if(member_==0) goto end;
if(p[fn][member_]=='\n') {member_++;break;}

member_--;
}

member_Start=member_;

while(1){
if(member==0) goto end;
if(p[fn][member]==' ' &&
   ((spacecheck(member_Start,member)==SPCNUM && spaces==1) ||
    p[fn][member-1]=='>')) break;

member--;
}

dm=member_Return-member-1;

strncpy(buf,&p[fn][member+1],dm);
buf[dm]='\0';
if(strlen(buf)==0) goto end;

getcwd(buf_,ASIZE);
if(buf_[/*3*/1]!='\0') strcat(buf_,"/");
strcat(buf_,buf);
```

```
strcpy(home,home_global);
strcat(home,"zzz.filename");

fpf=fopen(home,"ab");
fwrite(&buf_[0],1,strlen(buf_),fpf);
fwrite(&two[4][0],1,1,fpf);
fclose(fpf);

dx=0;j=ROW_L+2;dy=j*UDY;
cleardevice_(1,dx,dy,XRESO,UDY);
sprintf(mline,"  File = %s",
        /*linefrom1*/buf_);

ROWflag=1;
while_puts_show_monitorline(0,ACTIVE,j);
ROWflag=0;
bitblt(1,dx,dy,XRESO,UDY,dx,dy);

BitBltflag=2;

beep(50);

end:{}
}/** FILE_filename **/


void FILE_ref_tmp(char flag)        /* find, home */
{
char type;

tailcheck();

if(cut==0){
k_from=topp[/*firstline+*/jcsr]+0;
k_to=top_icsr(/*firstline+*/jcsr,return_is(/*firstline+*/jcsr));
if(k_to!=kmax[fn]){
type=gettype_p(k_to);
if(type<=2) k_to++;
else if(type==3) k_to+=2;
else ;}
}
else if(cut==1){
k_to=topp[/*firstline+*/jcsr]+0;
}
else{
```

```
k_to=top_icsr(/*firstline+*/jcsr,icsr);
}

if(k_to-k_from!=0){
if(cut>0) swap_BL(1);
dk_file=k_to-k_from;
if(flag==0)
text_to_file(1,0);
else{
d_or_t=1;
text_to_file((char)(2+cut),0);
d_or_t=0;
}

beep(50);
}

if(cut>0){
cut=0;
page_firstk/*_from*/(firstk);
/*csr();*/
}
}/** FILE_ref_tmp **/


void dlgproc_FILE(char flag_append)
{
char type;
int icsr_old,jcsr_old;
long firstk_old,k_from_old;
unsigned char oldstring[ASIZE];

if(cut==0) return;

if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

tailcheck();
strcpy(oldstring,file_SA);
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;
k_from_old=k_from;

while(1){
start:

dialogflag=1;
```

```
strcpy(p_dialog,file_SA);              /* to p_dialog */
if(flag_append==0) before_mainroop("Save (p)");
else before_mainroop("Save (p,a)");
mainroop();                           /* p_dialog */
after_mainroop();
strcpy(array,p_dialog);               /* to array */

if(dialogflag==3) break;

/*strcpy(file_old,array);*/
if(use_selector_flag==1) use_selector(1);
if((arraycheckflag=arraycheck())>1) use_filer();

/*printf_(arraycheckflag);*/

if(/*(arraycheckflag=*/arraycheck()/*)*/>0){
/*printf_(arraycheckflag);
puts_mline(0,array);puts_mline_flag=0;*/
/*file_SA[0]='\0';*/strcpy(file_SA,/*file_old*/oldstring);
if(strlen(array)>0) puts_mline(0,"The directory is bad.");}
else {strcpy(file_SA,array);break;}
}/**while(1)**/

if(dialogflag==3){
dialogflag=0;
strcpy(file_SA,oldstring);
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);

firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
/*cut=0;*/
page_firstk(firstk);
/*csr();*/
}/**if(dialogflag)**/

else if(dialogflag==2){                  /* job */
dialogflag=0;
fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);

firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
while_puts_show_(0,firstk);
```

```
if(cut==0){
k_from=topp[/*firstline_old+*/jcsr_old]+0;
k_to=top_icsr(/*firstline_old+*/jcsr_old,return_is(/*firstline_old+*/jcsr_old));
if(k_to!=kmax[fn]){
type=gettype_p(k_to);
if(type<=2) k_to++;
else if(type==3) k_to+=2;
else ;}
}
else if(cut==1){
k_from=k_from_old;
k_to=topp[/*firstline_old+*/jcsr_old]+0;
}
else{
k_from=k_from_old;
k_to=top_icsr(/*firstline_old+*/jcsr_old,icsr_old);
}

if(k_to-k_from!=0){
if(cut>0) swap_BL(1);
dk_file=k_to-k_from;
if(text_to_file(0,flag_append)==1){
strcpy(file_SA,/*file_old*/oldstring);
/*message(3,1);*/puts_mline(0,"Reinput a filename.");goto start;}
}

/*firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;*/

cut=0;
page_firstk(firstk);
/*csr();*/
}/**if(dialogflag)**/
}/** dlgproc_FILE **/


void refind(char flag)
{
if(function==1) return;
if(strlen(ref_s)==0) return;

if(flag==1) reffunc_REF=0;
else if(flag==2) reffunc_REF=1;

redoskip=1;
  nest_free();
```

```
tailcheck();BitBltflag=0;
}/** refind **/


void dlgproc_REF(char reffunc)
{
char BitBltflag_REF;
int icsr_old,jcsr_old;
long firstk_old;
unsigned char oldstring[ASIZE];

if(filerskip==1) {filerskip=0;reffunc=reffunc_REF;goto skip;}
if(redoskip==1) {redoskip=0;reffunc=reffunc_REF;goto skip;}

reffunc_REF=reffunc;
monitorline(1);
if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

tailcheck();
strcpy(oldstring,ref_s);
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;
BitBltflag_REF=1;

start:

dialogflag=1;
dialogflag_REF=1;

strcpy(p_dialog,ref_s);             /* to p_dialog */
if(u_s_flag){
u_s_flag=0;dialogflag=2;use_selector_flag=1;
before_mainroop_("Find");
trim_dialog();
}
else{
before_mainroop("Find");
mainroop();                         /* p_dialog */
}
after_mainroop();
strcpy(ref_s,p_dialog);

if(dialogflag==3){
dialogflag=0;
dialogflag_REF=0;
strcpy(ref_s,oldstring);
```

```
icsr=icsr_old;jcsr=jcsr_old;
/*BitBlt_full();*/page_firstk(firstk);
nestflag=0;
}

else if(dialogflag==2){                  /* job */
dialogflag=0;
dialogflag_REF=0;

/*strcpy(ref_s_old,ref_s);*/

if(strlen(ref_s)==0 || use_selector_flag==1){
if(fn!=FMAX-1 && cut==0){
use_selector(0);
strcpy(ref_s,array);

fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
firstk=firstk_old;                   /* read_3vals() */
icsr=icsr_old;jcsr=jcsr_old;

if(strlen(ref_s)==0){
restore_page_and_oldcsr();
strcpy(ref_s,/*ref_s_old*/oldstring);goto start;
}
else{
no_extraline=1;
page_firstk(firstk);
no_extraline=0;
csr();
BitBltflag_REF=0;
}
}/**if(fn,cut)**/
else{
goto start;
}/**else(fn,cut)**/
}

icsr=icsr_old;jcsr=jcsr_old;
if(BitBltflag_REF){
no_extraline=1;
/*BitBlt_full();*/page_firstk(firstk);
no_extraline=0;
csr();
}
```

```
skip:

reference(reffunc);

/*if(filerflag==1) icsr=0;*/            /* pending */
direction=0;
}
}/** dlgproc_REF **/


void dlgproc_REP(char reffunc)
{
char editflag_old,dialogflag_old;
int icsr_old,jcsr_old;
long firstk_old,kmax_old,k_from_old;
unsigned char *ptmp_rep;
unsigned char oldstring[ASIZE],oldstring_[ASIZE],ref_t_old[ASIZE];

if(editflag[fn]<=-1) {message(13,1);csr();return;}

reffunc_REP=reffunc;
monitorline(1);
if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

repcount=0;

tailcheck();
strcpy(oldstring,rep_s);
strcpy(oldstring_,rep_s_);
firstk_old=firstk;
icsr_old=icsr;jcsr_old=jcsr;

while(1){
start:

dialogflag=1;
dialogflag_REF=1;

strcpy(p_dialog,rep_s);             /* to p_dialog */
before_mainroop("Replace");
mainroop();                         /* p_dialog */
after_mainroop();
strcpy(rep_s,p_dialog);

if(dialogflag==3) break;
```

```
/*strcpy(rep_s_old,rep_s);*/

if(strlen(rep_s)==0 || use_selector_flag==1){
if(fn!=FMAX-1 && cut==0){
use_selector(0);
strcpy(rep_s,array);

fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
firstk=firstk_old;                  /* read_3vals() */
icsr=icsr_old;jcsr=jcsr_old;

if(strlen(rep_s)==0){
dialogflag_old=dialogflag;dialogflag=0;
restore_page_and_oldcsr();
dialogflag=dialogflag_old;
strcpy(rep_s,/*rep_s_old*/oldstring);goto start;
}
else{
page_firstk(firstk);
dialogflag_old=dialogflag;dialogflag=0;
csr();
dialogflag=dialogflag_old;
/*BitBltflag_REF=0*/;
}
}/**if(fn,cut)**/
else{
goto start;
}/**else(fn,cut)**/
}

start_:

dialogflag=2;

strcpy(p_dialog,rep_s_);            /* to p_dialog */
before_mainroop("with");
mainroop();                         /* p_dialog */
after_mainroop();
strcpy(rep_s_,p_dialog);

if(dialogflag==3){
strcpy(rep_s_,oldstring_);
}

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0)
```

```
goto start;

if(/*strlen(rep_s)==0 || */dialogflag!=3 && use_selector_flag==1){
if(fn!=FMAX-1 && cut==0){
use_selector(0);
strcpy(rep_s_,array);

fn=ftable[ftp-1].fn;
strcpy(fname,fnames[fn]);
firstk=firstk_old;                  /* read_3vals() */
icsr=icsr_old;jcsr=jcsr_old;

if(strlen(rep_s_)==0){
dialogflag_old=dialogflag;dialogflag=0;
restore_page_and_oldcsr();
dialogflag=dialogflag_old;
strcpy(rep_s_,/*rep_s_old*/oldstring);goto start_;
}
else{
/*page_firstk(firstk);
dialogflag_old=dialogflag;dialogflag=0;
csr();
dialogflag=dialogflag_old;*/
/*BitBltflag_REF=0*/;
}
}/**if(fn,cut)**/
else{
goto start_;
}/**else(fn,cut)**/
}

if(dialogflag==2) break;
}/**while(1)**/

if(dialogflag==3){
dialogflag=0;
dialogflag_REF=0;
strcpy(rep_s,oldstring);
icsr=icsr_old;jcsr=jcsr_old;
/*BitBlt_full();*/page_firstk(firstk);
}

else if(dialogflag==2){                  /* job */
dialogflag=0;
dialogflag_REF=0;
icsr=icsr_old;jcsr=jcsr_old;
```

```
/*imm_pause();*/

if(cut==0){
menuflag=2;
mnuproc_REP("Range");
if(refill==0){
icsr=icsr_old;jcsr=jcsr_old;
imm_restart();

refill=1;
if(/*divideflag==1*/0) restore_another();
page_firstk(firstk);
extraline(1);goto end;}

if(jcsr_select==1) {allflag=1;direction=1;}
else if(jcsr_select==2) {allflag=0;direction=1;}
else {allflag=0;direction=0;}
}/**if(cut)**/
else{
                            /* from before_mainroop_menu_REP() */
cleardevice_(-1,0,0,0,0);
while_puts_show_(1,firstk);         /* erase the dialog window */
monitorline(0);
menuflag=0;

if(dialogflag==0 && menuflag==0 && filerflag==0){
if(cut>0) indicator(1);
else {if(indicationflag) {indicationflag=0;indicator(0);}}
}
else BitBlt_indicator();
if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

page_firstk(firstk);csr();


if(cut==1) {k_to=topp[jcsr]+0;if(k_to-k_from>0) k_to--;}
else if(cut==2) {k_to=top_icsr(jcsr,icsr);if(k_to-k_from>0) k_to--;}

if(k_to-k_from<0) {allflag=0;direction=1;}
else if(k_to-k_from>0) {allflag=0;direction=0;}
else {page_firstk(firstk);goto end;}
}/**else(cut)**/

if(cut==0) menuflag=0;
```

```
puts_mline(0,"Do you replace all ? (Y/N)");/*puts_mline_flag=1;*/
while(1){
yorn=subroop();
if(yorn<=2) break;
}

/*if(cut==0) menuflag=0;*/
icsr=icsr_old;jcsr=jcsr_old;

/*no_extraline=1;*/
if(/*cut==0 && divideflag==1*/0) restore_another();

if(yorn==0)      lumpflag=1;        /* Y */
else if(yorn==1) lumpflag=0;        /* N */
else{                               /* Esc, Pause */
/*no_extraline=0;*/
/*if(cut>0) cut=0;*/
imm_restart();
page_firstk(firstk);goto end;
}

ptmp_rep=(unsigned char *)malloc(sizeof(unsigned char)*(kmax[fn]+(1+1)));

if(ptmp_rep!=NULL){
memcpy(&ptmp_rep[0],&p[fn][0],kmax[fn]+1);
kmax_old=kmax[fn];editflag_old=editflag[fn];k_from_old=k_from;

strcpy(ref_t_old,ref_t);
shorten_();

no_extraline=1;
if(lumpflag==1){
lumpflag=0;
page_firstk(firstk);/*csr();*/
lumpflag=1;
reference_lump(reffunc);
}
else
reference(reffunc);
no_extraline=0;

if(cut>0) cut=0;

strcpy(ref_t,ref_t_old);
imm_restart();
```

```
if(flag_global){
flag_global=0;
message(7,2);

memcpy(&p[fn][0],&ptmp_rep[0],kmax[fn]+1);
kmax[fn]=kmax_old;/*linemax[fn]=while_puts_fload(1);*/editflag[fn]=editflag_old;
k_from=k_from_old;

repcount=0;
firstk=firstk_old;
icsr=icsr_old;jcsr=jcsr_old;
page_firstk(firstk);
}

direction=0;
puts_mline(1,"string(s) replaced.");/*puts_mline_flag=1;*/
csr();BitBltflag=/*1*/2;
free(ptmp_rep);
}/**if(ptmp_rep)**/
else{
/*no_extraline=0;*/
/*if(cut>0) cut=0;*/
imm_restart();
message(7,2);
allflag=0;lumpflag=0;
/*BitBlt_full();*/page_firstk(firstk);
}/**else(ptmp_rep)**/

end:{}
}
}/** dlgproc_REP **/

/************************** dialog functions -> ***************************/

void backspace_dialog(void)
{
char flag;
long k,k_icsr,firstk_dialog_;

flag=csr_left_dialog();

if(flag!=1){
lumpflag_dialog=1;
deletion_dialog();
lumpflag_dialog=0;
```

```
if(flag==2){                        /* scrolled up */
k_icsr=firstk_dialog+icsr;

if(firstk_dialog-(CD-1)>0){
firstk_dialog_=max(min(firstk_dialog-(CD-1),kmax_dialog),0);  /* protection */
firstk_dialog=gethead_dialog(1,firstk_dialog_);    /* for jp */
page_firstk_dialog(firstk_dialog);
icsr=k_icsr-firstk_dialog;
}
else{
page_firstk_dialog(0);
icsr=k_icsr-0;
}
}/**if(flag)**/
else{
page_firstk_dialog(firstk_dialog);
}/**else(flag)**/
}/**if(flag!)**/
}/** backspace_dialog **/


int insertion_dialog(unsigned char charcode)
{
char flag_;
long k;

tailcheck_dialog();

flag_=0;

kmax_dialog++;
if(kmax_dialog>ASIZEM-1) {beep(50);kmax_dialog--;flag_=1;}
else{
k=firstk_dialog+icsr;
memmove(&p_dialog[k+1],&p_dialog[k],kmax_dialog-1-k+1);
p_dialog[k]=charcode;
}

page_firstk_dialog(firstk_dialog);

if(flag_==0){
csr_right_dialog();
if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}  /* for bad */
return 0;
}
else{
```

```
return 1;
}
}/** insertion_dialog **/


int deletion_dialog(void)
{
char type;
long k,dk;

tailcheck_dialog();

k=firstk_dialog+icsr;
if(k==kmax_dialog) return 1;

type=gettype_dialog(k);

if(type<=2){
dk=1;
memmove(&p_dialog[k],&p_dialog[k+dk],kmax_dialog-(k+dk)+1);
kmax_dialog-=dk;
}/**if(type)**/
else if(type==3){
dk=2;
memmove(&p_dialog[k],&p_dialog[k+dk],kmax_dialog-(k+dk)+1);
kmax_dialog-=dk;
}/**else if(type)**/
else{
}/**else(type)**/

if(overwriteflag==1) return 1;

page_firstk_dialog(firstk_dialog);
if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}

return 0;
}/** deletion_dialog **/


void while_puts_show_dialog(long k)
{
char TextOutflag,type;
int i,j,dx,dy;
unsigned char s[1],s_[1];
unsigned char jis[2];
```

```
TextOutflag=1;
i=0;j=0;

while(1){
s[0]=p_dialog[k];
type=gettype_dialog(k);

if(type<=2){
if(TextOutflag){
if(s[0]>=0x20 && type==0)
setstccolor(bfset[WB].fore);
else if(type==-1)
setstccolor(/*12*/bfset[WB].fore);
/*else if(s[0]==0x1a)*/
else if(k==kmax_dialog)
setstccolor(12);
/*else if(s[0]=='\n')
setstccolor(RETURN);*/
/*else if(s[0]==0x09)
setstccolor(TABCOLOR);*/
else
setstccolor(/*CC*/RTC);

dx=(i+DI_d)*UDX;dy=(j+DJ_d)*UDY;

if(s[0]>=0x20 && type==0)
stc(1,dx,dy,s,1);
/*else if(s[0]=='\n'){
s_[0]=0x0d;
stc(1,dx,dy,s_,1);
}*/
/*else if(s[0]==0x1a){*/
else if(k==kmax_dialog){
s_[0]=/*0x0d*/dummy_R;
stc(1,dx,dy,s_,1);
}
else if(type==-1){
s_[0]=0x0d;
stc(1,dx,dy,s_,1);
}
/*else if(s[0]==0x09){
s_[0]=0x0d;
stc(1,dx,dy,s_,1);
}*/
else{
if(s[0]==0x7f) s[0]=0x00;
```

```
s_[0]=cc[s[0]];
stc(1,dx,dy,s_,1);
}
}/**if(TextOutflag)**/

/*if(k==kmax_dialog) break;*/

k++;
i++;
if(i==CD) break;
}/**if(type)**/
else if(type==3){
if(TextOutflag){
jis[0]=p_dialog[k];
jis[1]=p_dialog[k+1];

dx=(i+DI_d)*UDX;dy=(j+DJ_d)*UDY;
setstccolor(bfset[WB].fore);
stc(1,dx,dy,jis,2);
}/**if(TextOutflag)**/

/*if(k==kmax_dialog) break;*/              /* ? */

k+=2;
i+=2;
if(i>=CD) break;
}/**else if(type)**/
else{
}/**else(type)**/

if(k>kmax_dialog) break;            /* new break */
}
}/** while_puts_show_dialog **/


void text_end_dialog(void)
{
long firstk_dialog_;

firstk_dialog_=max(/*min(*/kmax_dialog-(CD-1)/*,kmax_dialog)*/,0);  /* protection */
firstk_dialog=gethead_dialog(1,firstk_dialog_);    /* for jp */

page_firstk_dialog(firstk_dialog);
csr_row_end_dialog();
}/** text_end_dialog **/
```

```
void page_down_dialog(void)
{
long firstk_dialog_;

if(firstk_dialog+CD-1+icsr<=kmax_dialog)
firstk_dialog_=max(min(firstk_dialog+CD-1,kmax_dialog),0);  /* protection */
else
firstk_dialog_=max(/*min(*/kmax_dialog-icsr/*,kmax_dialog)*/,0);  /* protection */
firstk_dialog=gethead_dialog(1,firstk_dialog_);    /* for jp */

page_firstk_dialog(firstk_dialog);
/*within_linemax_dialog();*/
tailcheck_dialog();
}/** page_down_dialog **/


void page_up_dialog(void)
{
long firstk_dialog_;

firstk_dialog_=max(min(firstk_dialog-(CD-1),kmax_dialog),0);  /* protection */
firstk_dialog=gethead_dialog(1,firstk_dialog_);    /* for jp */

page_firstk_dialog(firstk_dialog);
/*within_linemax_dialog();*/
tailcheck_dialog();
}/** page_up_dialog **/


void trim_dialog(void)
{
p_dialog[kmax_dialog]='\0';
}/** trim_dialog **/


void restore_dialog(void)
{
strcpy(p_dialog,p_restore);

kmax_dialog=strlen(p_dialog)-1;
text_end_dialog();

if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}  /* for bad */
}/** restore_dialog **/
```

```
void clear_dialog(char flag)
{
if(driveflag) return;

kmax_dialog=0;
p_dialog[0]=0x1a;
p_dialog[1]='\0';

/*within_linemax_dialog();*/
tailcheck_dialog();
page_firstk_dialog(firstk_dialog);

if(flag) {if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}}
}/** clear_dialog **/


void page_firstk_dialog(long k)
{
int i,j,dx,dy;

firstk_dialog=max(min(k,kmax_dialog),0);   /* protection */

if(lumpflag_dialog==1) return;
if(dbflag==1) return;

i=DI_d;j=DJ_d;dx=i*UDX;dy=j*UDY;    /* small */
cleardevice_(1,dx,dy,UDX*(CD+1),UDY);
while_puts_show_dialog(firstk_dialog);

BitBlt_dialog();
}/** page_firstk_dialog **/


void page_firstk_dialog_dummy(long k)
{
kmax_dialog=0;

while_puts_show_dialog(/*firstk_dialog*/0);
}/** page_firstk_dialog_dummy **/


void BitBlt_dialog(void)
{
int i,j,dx,dy;
```

```
i=DI_d-1;j=DJ_d-1;dx=i*UDX;dy=j*UDY;    /* large */
bitblt(1,dx,dy,UDX*((CD+1)+2),UDY*(1+2),dx,dy);

BitBltflag_=1;
}/** BitBlt_dialog **/


void tailcheck_dialog(void)
{
within_linemax_dialog();

csr_tab_dialog(0);
}/** tailcheck_dialog **/


void within_linemax_dialog(void)
{
/*if(firstk_dialog>kmax_dialog) firstk_dialog=kmax_dialog;*/
firstk_dialog=max(min(firstk_dialog,kmax_dialog),0);   /* protection */

if(firstk_dialog+icsr>kmax_dialog) icsr=kmax_dialog-firstk_dialog;
}/** within_linemax_dialog **/


void csr_row_home_dialog(void)
{
icsr=0;
}/** csr_row_home_dialog **/


void csr_row_end_dialog(void)
{
icsr=CD-1;

/*within_linemax_dialog();*/
tailcheck_dialog();
}/** csr_row_end_dialog **/


char csr_left_dialog(void)
{
icsr--;
within_linemax_dialog();

if(icsr<0){
icsr=0;
```

```
if(scroll_up_dialog()==1)
return 1;
else{
/*csr_tab_dialog(0);*/
return 2;}
}

csr_tab_dialog(0);

return 0;
}/** csr_left_dialog **/


void csr_right_dialog(void)
{
char type;
long k,k_icsr;

within_linemax_dialog();
k=firstk_dialog+icsr;
if(k==kmax_dialog) return;

icsr++;
csr_tab_dialog(1);
k_icsr=firstk_dialog+icsr;

while(1){
if(icsr>CD-1){
scroll_down_dialog();
icsr=k_icsr-firstk_dialog;}
else break;
}
}/** csr_right_dialog **/


int scroll_down_dialog(void)
{
if(firstk_dialog>=kmax_dialog) return 1;

firstk_dialog++;
firstk_dialog=gethead_dialog(1,firstk_dialog);    /* for jp */
page_firstk_dialog(firstk_dialog);

/*within_linemax_dialog();*/
tailcheck_dialog();
```

```
return 0;
}/** scroll_down_dialog **/


int scroll_up_dialog(void)
{
if(firstk_dialog<1) return 1;

firstk_dialog--;
firstk_dialog=gethead_dialog(0,firstk_dialog);    /* for jp */
page_firstk_dialog(firstk_dialog);

/*within_linemax_dialog();*/
tailcheck_dialog();

return 0;
}/** scroll_up_dialog **/

/************************** <- dialog functions ***************************/

int GKS(KeySym XK)
{
if(keysym==XK) return -1;
else return 0;
}/** GKS **/


int GKS_(long ModkeyMask)
{
if((event.xkey.state & ModkeyMask)>0) return -1;
else return 0;
}/** GKS_ **/


void kbhit_(void)
{
start:

XNextEvent(d,&event);
if(XFilterEvent(&event,None)){      /* for IM Server */
/*if(event.type==KeyRelease){
while(1){
if(XEventsQueued(d,QueuedAlready)<=QMAX) break;
else XNextEvent(d,&event_);
```

```
if(event_.type!=KeyPress && event_.type!=KeyRelease){
event=event_;
break;}
}
}*/

goto start;
}

/*if(event.type==KeyRelease){
while(1){
if(XEventsQueued(d,QueuedAlready)<=QMAX) break;
else XNextEvent(d,&event_);

if(event_.type!=KeyPress && event_.type!=KeyRelease){
event=event_;
break;}
}
}*/

if(event.type==EnterNotify){
/*if(menuflag>0){}
else if(filerflag==1 && dialogflag==0){}
else imm_restart();*/               /* <ble204> */
goto start;
}
else if(event.type==LeaveNotify){
/*imm_pause();*/                    /* <ble204> */
goto start;
}
else{
if(cut>0) wndproc_BL();
else if(filerflag==1) wndproc_filer();
else if(deletedflag==1) wndproc_deleted();
else if(refflag==1) wndproc_ref();
else wndproc();
}
}/** kbhit_ */


int wndproc_filer(void)
{
char gotoflag;
int length;
/*static */char buf[10];
/*static */unsigned char buf_Xmb[ASIZE];
```

```
entrance:
/*length=*/XLookupString((XKeyEvent *)&event,buf,10,&keysym,NULL);
/*buf[length]='\0';*/

length=XmbLookupString(ic,(XKeyEvent *)&event,buf_Xmb,ASIZE,&sym,&st);
if(st==XLookupBoth || st==XLookupChars){}
else length=0;
if(length>0) buf_Xmb[length]='\0';

/*if(event.type==KeyRelease){
if(dialogflag>0) InputPosition(ic,icsr,jcsr);
}*/
if(dialogflag>0) imm_restart();

if(event.type==KeyPress){
BitBltflag=0;BitBltflag_=0;

/************************** menu keydowns -> ***************************/
/************************** <- menu keydowns ***************************/

/************************** dialog keydowns -> ***************************/

if(dialogflag>0){

if(usflag==1) usflag=0;

if(cqflag==2){
  BitBltflag_=2;
  goto end_dialog;}
if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_dialog;}

gotoflag=1;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  imm_pause();
  dialogflag=3;refill=0;BitBltflag_=2;}
else if(GKS(XK_Return)<0){
  imm_pause();
  trim_dialog();
  dialogflag=2;refill=0;BitBltflag_=2;}

else {gotoflag=0;}
```

```
if(gotoflag==1) goto end_dialog;

end_left_dialog:
left_keydowns_dialog();

end_dialog:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                    /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag_==0) {BitBlt_dialog();csr();}
else if(BitBltflag_==1)             csr();
else{}

return 1;
}/**if(dialogflag)**/

/************************** <- dialog keydowns ****************************/

if(function==2){
keydowns_f2();

return 1;
}

if(usflag==1) usflag=0;

if(cqflag==6){
  /*BitBltflag=2;*/
  goto end_left;}

gotoflag=1;

/*9*/
if(GKS_(ShiftMask)<0 && GKS(XK_F4)<0){
  if(function==1) filerskip=1;
  refill=-2;/*if(GKS_(ShiftMask)<0) refill--;*/charflag=0;charcode=2;
  filer_execute=1;
  BitBltflag=2;}
```

```
else if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  if(function==0){
  refill=0;if(GKS_(ShiftMask)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}
else{
  charflag=0;charcode=2;
  BitBltflag=2;}}
else if(GKS(XK_Return)<0){
  if(function==1) filerskip=1;
  refill=-2;if(GKS_(ShiftMask)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}

else if(GKS_(ControlMask)<0 && (GKS('Q')<0 || GKS('q')<0)){
  cqflag=5;prompt_cq(2);}

else if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)<0 && (GKS('F')<0 || GKS('f')<0)){
  FILE_filename();}

else if(GKS(XK_F9)<0){
  Find(0);}

else {gotoflag=0;}

if(gotoflag==1) goto end;

end_left:
gotoflag=left_keydowns();

if(gotoflag==0 && cqflag>0) goto end_;  /* no job */

end:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                    /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(topp[jcsr]<topp_floor) {icsr=0;jcsr=jcsr_floor/*+1*/;page_firstk(0);}
if(jcsrmax==0) {scroll_up(0);} if(jcsr>jcsrmax-1) jcsr=jcsrmax-1;

if(BitBltflag==0)      {BitBlt_full();csr();}
else if(BitBltflag==1) {monitorline(1);csr();}
```

```
else{}

end_:
return 1;
}/**if(event.type)**/
else if(event.type==Expose){
restore_in_PAINT();

return 1;
}/**else if(event.type)**/
else{}

return 0;
}/** wndproc_filer **/


int wndproc_ref(void)
{
char gotoflag;
int length;
/*static */char buf[10];
/*static */unsigned char buf_Xmb[ASIZE];

entrance:
/*length=*/XLookupString((XKeyEvent *)&event,buf,10,&keysym,NULL);
/*buf[length]='\0';*/

length=XmbLookupString(ic,(XKeyEvent *)&event,buf_Xmb,ASIZE,&sym,&st);
if(st==XLookupBoth || st==XLookupChars){}
else length=0;
if(length>0) buf_Xmb[length]='\0';

InputPosition(ic,icsr,jcsr);

if(event.type==KeyPress){
BitBltflag=0;

/************************** menu keydowns -> ****************************/
/************************** <- menu keydowns ****************************/

/************************** dialog keydowns -> ****************************/
/************************** <- dialog keydowns ****************************/

if(function==2){
keydowns_f2();
```

```
return 1;
}

if(usflag==1) usflag=0;

if(cqflag==2){
  BitBltflag=2;
  goto end;}
if(cqflag==6){
  /*BitBltflag=2;*/
  goto end_left;}

gotoflag=1;

/*9*/
if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  refill=0;if(GKS_(ShiftMask)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}
/* 4if */
else if(GKS(XK_Return)<0){
  if(GKS_(ShiftMask)<0 || GKS_(ControlMask)<0){
  if(Enter(1)==1) goto end;}
  else{
  refill=-2;charflag=0;charcode=2;
  BitBltflag=2;}}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('I')<0 || GKS('i')<0)){
  cqflag=1;prompt_cq(0);}
else if(GKS_(ControlMask)<0 && (GKS('Q')<0 || GKS('q')<0)){
  cqflag=5;prompt_cq(2);}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('Y')<0 || GKS('y')<0)){
  YKP_word(0);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('K')<0 || GKS('k')<0)){
  YKP_word(1);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('P')<0 || GKS('p')<0)){
  YKP_word(2);}
else if(GKS_(ControlMask)<0 && (GKS('Y')<0 || GKS('y')<0)){
  YKP(0);}
else if(GKS_(ControlMask)<0 && (GKS('K')<0 || GKS('k')<0)){
  YKP(1);}
else if(GKS_(ControlMask)<0 && (GKS('P')<0 || GKS('p')<0)){
  YKP(2);}

else if(GKS(XK_F6)<0){
  paste=0;cut=0;monitorline(1);BitBltflag=2;}
```

```
else if(GKS_(ShiftMask)<0 && GKS(XK_F7)<0){
  paste=2;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS(XK_F7)<0){
  paste=1;cut=0;monitorline(1);BitBltflag=2;}

else {gotoflag=0;}

if(gotoflag==1) goto end;

end_left:
gotoflag=left_keydowns();

if(gotoflag==0 && cqflag>0) goto end_;  /* no job */

end:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                   /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag==0)      {BitBlt_full();csr();}
else if(BitBltflag==1) {monitorline(1);csr();}
else{}

end_:
return 1;
}/**if(event.type)**/
else if(event.type==Expose){
restore_in_PAINT();

return 1;
}/**else if(event.type)**/
else{}

return 0;
}/** wndproc_ref **/


int wndproc_deleted(void)
{
char gotoflag;
```

```
int length;
/*static */char buf[10];
/*static */unsigned char buf_Xmb[ASIZE];

entrance:
/*length=*/XLookupString((XKeyEvent *)&event,buf,10,&keysym,NULL);
/*buf[length]='\0';*/

length=XmbLookupString(ic,(XKeyEvent *)&event,buf_Xmb,ASIZE,&sym,&st);
if(st==XLookupBoth || st==XLookupChars){}
else length=0;
if(length>0) buf_Xmb[length]='\0';

InputPosition(ic,icsr,jcsr);

if(event.type==KeyPress){
BitBltflag=0;BitBltflag_=0;

/************************* menu keydowns -> ****************************/

if(menuflag>0){

if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_menu;}

gotoflag=1;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  menuflag=3;refill=0;BitBltflag_=2;}
else if(GKS(XK_Return)<0){
  menuflag=2;refill=0;BitBltflag_=2;}

else {gotoflag=0;}

if(gotoflag==1) goto end_menu;

end_left_menu:
left_keydowns_menu();

end_menu:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                    /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
```

```
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag_==0) {BitBlt_menu();csr();}
else if(BitBltflag_==1) csr();
else{}

return 1;
}/**if(menuflag)**/

/************************** <- menu keydowns ****************************/

/************************** dialog keydowns -> ****************************/

if(dialogflag>0){

if(usflag==1) usflag=0;

if(cqflag==2){
  BitBltflag_=2;
  goto end_dialog;}
if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_dialog;}

gotoflag=1;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  dialogflag=3;refill=0;BitBltflag_=2;}
else if(GKS(XK_Return)<0){
  trim_dialog();
  dialogflag=2;refill=0;BitBltflag_=2;}

else {gotoflag=0;}

if(gotoflag==1) goto end_dialog;

end_left_dialog:
left_keydowns_dialog();

end_dialog:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                  /* double byte */
```

```
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag_==0) {BitBlt_dialog();csr();}
else if(BitBltflag_==1)             csr();
else{}

return 1;
}/**if(dialogflag)**/

/************************** <- dialog keydowns ****************************/

if(function==2){
keydowns_f2();

return 1;
}

if(usflag==1) usflag=0;
if(function==3) function=0;
if(function==4) function=1;

if(cqflag==2){
  BitBltflag=2;
  goto end;}
if(cqflag==6){
  /*BitBltflag=2;*/
  goto end_left;}

gotoflag=1;

/*9*/
if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  if(function==0){
  refill=0;if(GKS_(ShiftMask)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}
else{
  charflag=0;charcode=2;
  BitBltflag=2;}}
else if(GKS(XK_Return)<0){
  if(Enter(0)==1) goto end;}
/* 4if */
```

```
/*else if(GKS(XK_Return)<0){
  if(GKS_(ShiftMask)<0 || GKS_(ControlMask)<0){
  if(Enter(1)==1) goto end;}
  else{
  refill=-2;charflag=0;charcode=2;
  BitBltflag=2;}}*/

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('I')<0 || GKS('i')<0)){
  cqflag=1;prompt_cq(0);}
else if(GKS_(ControlMask)<0 && (GKS('Q')<0 || GKS('q')<0)){
  cqflag=5;prompt_cq(2);}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('Y')<0 || GKS('y')<0)){
  YKP_word(0);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('K')<0 || GKS('k')<0)){
  YKP_word(1);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('P')<0 || GKS('p')<0)){
  YKP_word(2);}
else if(GKS_(ControlMask)<0 && (GKS('Y')<0 || GKS('y')<0)){
  YKP(0);}
else if(GKS_(ControlMask)<0 && (GKS('K')<0 || GKS('k')<0)){
  YKP(1);}
else if(GKS_(ControlMask)<0 && (GKS('P')<0 || GKS('p')<0)){
  YKP(2);}

else if(GKS(XK_F6)<0){
  paste=0;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS_(ShiftMask)<0 && GKS(XK_F7)<0){
  paste=2;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS(XK_F7)<0){
  paste=1;cut=0;monitorline(1);BitBltflag=2;}

else if(GKS(XK_F5)<0){
  Replace();}
else if(GKS(XK_F9)<0){
  Find(0);}

else {gotoflag=0;}

if(gotoflag==1) goto end;

end_left:
gotoflag=left_keydowns();

if(gotoflag==0 && cqflag>0) goto end_;  /* no job */
```

```
end:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                    /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag==0)      {BitBlt_full();csr();}
else if(BitBltflag==1) {monitorline(1);csr();}
else{}

end_:
return 1;
}/**if(event.type)**/
else if(event.type==Expose){
restore_in_PAINT();

return 1;
}/**else if(event.type)**/
else{}

return 0;
}/** wndproc_deleted **/


int wndproc_BL(void)
{
char gotoflag;
int length;
/*static */char buf[10];
/*static */unsigned char buf_Xmb[ASIZE];

entrance:
/*length=*/XLookupString((XKeyEvent *)&event,buf,10,&keysym,NULL);
/*buf[length]='\0';*/

length=XmbLookupString(ic,(XKeyEvent *)&event,buf_Xmb,ASIZE,&sym,&st);
if(st==XLookupBoth || st==XLookupChars){}
else length=0;
if(length>0) buf_Xmb[length]='\0';

InputPosition(ic,icsr,jcsr);
```

```
if(event.type==KeyPress){
BitBltflag=0;BitBltflag_=0;

/************************* menu keydowns -> ***************************/

if(menuflag>0){

if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_menu;}

gotoflag=1;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  menuflag=3;refill=0;BitBltflag_=2;}
else if(GKS(XK_Return)<0){
  menuflag=2;refill=0;BitBltflag_=2;}

else {gotoflag=0;}

if(gotoflag==1) goto end_menu;

end_left_menu:
left_keydowns_menu();

end_menu:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                    /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag_==0) {BitBlt_menu();csr();}
else if(BitBltflag_==1) csr();
else{}

return 1;
}/**if(menuflag)**/

/************************* <- menu keydowns ***************************/
```

```
/************************** dialog keydowns -> ****************************/

if(dialogflag>0){

if(usflag==1) usflag=0;

if(cqflag==2){
  BitBltflag_=2;
  goto end_dialog;}
if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_dialog;}

gotoflag=1;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  dialogflag=3;refill=0;BitBltflag_=2;
  /*if(GKS_(ShiftMask)<0) nocloseflag=1;
  else passflag=1;*/}
else if(GKS(XK_Return)<0){
  trim_dialog();
  if(GKS_(/*ControlMask*/ShiftMask)<0) use_selector_flag=1;else use_selector_flag=0;
  dialogflag=2;refill=0;BitBltflag_=2;}

else {gotoflag=0;}

if(gotoflag==1) goto end_dialog;

end_left_dialog:
left_keydowns_dialog();

end_dialog:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                    /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag_==0) {BitBlt_dialog();csr();}
else if(BitBltflag_==1)             csr();
else{}
```

```
return 1;
}/**if(dialogflag)**/

/************************* <- dialog keydowns ***************************/

if(function==2){
keydowns_f2();

return 1;
}

if(usflag==1) usflag=0;
if(function==3) function=0;
if(function==4) function=1;

if(cqflag==2){
  BitBltflag=2;
  goto end;}
if(cqflag==6){
  /*BitBltflag=2;*/
  goto end_left;}  /* to left_keydowns() */

gotoflag=1;

if(cqflag==4){        /* Esc-> */
if(GKS('S')<0 || GKS('s')<0){
  WM_func_CHAR(0);
  if(fn!=FMAX-1) dlgproc_FILE(0);length=0;}
else if(GKS('A')<0 || GKS('a')<0){
  WM_func_CHAR(0);
  if(fn!=FMAX-1) dlgproc_FILE(1);length=0;}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==4*/gotoflag==-1){
  if(GKS(XK_F12)<0 || GKS(XK_Pause)<0 || GKS(XK_F1)<0) cqflag=0;  /* Esc+ */
  BitBltflag=/*2*/1;
  goto end;}

if(gotoflag==1) goto end;else gotoflag=1;

/*9*/
if(GKS(XK_F2)<0){
  if(fn!=FMAX-1){
```

```
  if(GKS_(ShiftMask)<0) u_s_flag=1;else u_s_flag=0;
  dlgproc_JUMP();}}

else if(GKS(XK_Return)<0){
  if(Enter(0)==1) goto end;}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('I')<0 || GKS('i')<0)){
  cqflag=1;prompt_cq(0);}
else if(GKS(XK_Escape)<0){
  if(fn!=FMAX-1) {cqflag=3;prompt_cq(1);}}
else if(GKS(XK_F12)<0){  /* added */
  if(fn!=FMAX-1) {cqflag=3;prompt_cq(1);cqflag++;}}
else if(GKS_(ControlMask)<0 && (GKS('Q')<0 || GKS('q')<0)){
  cqflag=5;prompt_cq(2);}

else if(GKS_(ControlMask)<0 && (GKS('Y')<0 || GKS('y')<0)){
  YKP(0);}
else if(GKS_(ControlMask)<0 && (GKS('K')<0 || GKS('k')<0)){
  YKP(1);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('M')<0 || GKS('m')<0)){/* added */
  YKP(1);
  charflag=0;charcode=2;BitBltflag=/*2*/1;
  Quick_Find(2,1);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('N')<0 || GKS('n')<0)){/* added */
  YKP(1);
  charflag=0;charcode=2;BitBltflag=/*2*/1;
  Quick_Find(2,2);}
else if(GKS_(ControlMask)<0 && GKS('2')<0){
  if(GKS_(Mod1Mask)<0) FILE_jump(1);else FILE_jump(0);}
else if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)<0 && GKS('8')<0){ /* to zzz.string */
  if(fn!=FMAX-1) FILE_ref_tmp(1);}
else if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)<0 && GKS('9')<0){ /* to zzz.find */
  if(fn!=FMAX-1) FILE_ref_tmp(0);}

else if(GKS(XK_F6)<0){
  paste=0;cut=0;/*monitorline(1);BitBltflag=2;*/}
else if(GKS_(ShiftMask)<0 && GKS(XK_F7)<0){
  paste=2;cut=0;/*monitorline(1);BitBltflag=2;*/}
else if(GKS(XK_F7)<0){
  paste=1;cut=0;/*monitorline(1);BitBltflag=2;*/}

else if(GKS(XK_F8)<0){
  cut=0;/*monitorline(1);BitBltflag=2;*/}

else if(GKS(XK_F5)<0 && refflag==0){
  Replace();}
```

```
else if(GKS(XK_F9)<0 && refflag==0){
  Find(2);}

else {gotoflag=0;}

if(gotoflag==1) goto end;

end_left:
gotoflag=left_keydowns();

if(gotoflag==0 && cqflag>0) goto end_;  /* no job */

end:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                    /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}
if(fn!=FMAX-1 && ftp>0) write_3vals(ftp-1);

if(BitBltflag==0)      {BitBlt_full();csr();}
else if(BitBltflag==1) {monitorline(1);csr();}
else{}

/*no_extraline=0;*/

end_:
return 1;
}/**if(event.type)**/
else if(event.type==Expose){
restore_in_PAINT();

return 1;
}/**else if(event.type)**/
else{}

return 0;
}/** wndproc_BL **/


int wndproc(void)
{
```

```
char gotoflag;
int length;
/*static */char buf[10];
/*static */unsigned char buf_Xmb[ASIZE];

entrance:
/*length=*/XLookupString((XKeyEvent *)&event,buf,10,&keysym,NULL);
/*buf[length]='\0';*/

length=XmbLookupString(ic,(XKeyEvent *)&event,buf_Xmb,ASIZE,&sym,&st);
if(st==XLookupBoth || st==XLookupChars){}
else length=0;
if(length>0) buf_Xmb[length]='\0';

InputPosition(ic,icsr,jcsr);

if(event.type==KeyPress){
BitBltflag=0;BitBltflag_=0;

/************************* menu keydowns -> ****************************/

if(menuflag>0){

if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_menu;}

gotoflag=1;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  menuflag=3;refill=0;BitBltflag_=2;}
else if(GKS(XK_Return)<0){
  menuflag=2;refill=0;BitBltflag_=2;}

else {gotoflag=0;}

if(gotoflag==1) goto end_menu;

end_left_menu:
left_keydowns_menu();

end_menu:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                    /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
```

```
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag_==0) {BitBlt_menu();csr();}
else if(BitBltflag_==1) csr();
else{}

return 1;
}/**if(menuflag)**/

/************************** <- menu keydowns ****************************/

/************************** dialog keydowns -> ****************************/

if(dialogflag>0){

if(usflag==1) usflag=0;

if(cqflag==2){
  BitBltflag_=2;
  goto end_dialog;}
if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_dialog;}

gotoflag=1;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  dialogflag=3;refill=0;BitBltflag_=2;
  if(GKS_(ShiftMask)<0) nocloseflag=1;
  else passflag=1;}
else if(GKS(XK_Return)<0){
  trim_dialog();
  if(GKS_(/*ControlMask*/ShiftMask)<0) use_selector_flag=1;else use_selector_flag=0;
  dialogflag=2;refill=0;BitBltflag_=2;}

else {gotoflag=0;}

if(gotoflag==1) goto end_dialog;

end_left_dialog:
left_keydowns_dialog();
```

```
end_dialog:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                   /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag_==0) {BitBlt_dialog();csr();}
else if(BitBltflag_==1)             csr();
else{}

return 1;
}/**if(dialogflag)**/

/************************** <- dialog keydowns ****************************/

if(function==2){
keydowns_f2();

return 1;
}

if(usflag==1) usflag=0;
if(function==3) function=0;
if(function==4) function=1;

if(cqflag==2){
  BitBltflag=2;
  goto end;}
if(cqflag==6){
  /*BitBltflag=2;*/
  goto end_left;}  /* to left_keydowns() */
if(cqflag==8){
  /*BitBltflag=2;*/
  goto end_left;}  /* to left_keydowns() */

gotoflag=1;

if(cqflag==4){        /* Esc-> */
if(GKS('O')<0 || GKS('o')<0){
  WM_func_CHAR(0);
  open_file(0);length=0;}
```

```
else if(GKS('M')<0 || GKS('m')<0){
  WM_func_CHAR(0);
  open_file(1);length=0;}
else if(GKS('N')<0 || GKS('n')<0){
  WM_func_CHAR(0);
  open_file(2);length=0;}
else if(GKS('Q')<0 || GKS('q')<0 || GKS('U')<0 || GKS('u')<0){
  WM_func_CHAR(0);
  close_all();length=0;}
else if(GKS('W')<0 || GKS('w')<0 || GKS('Y')<0 || GKS('y')<0){
  WM_func_CHAR(0);
  if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)<0) end_ble();
  else save_all(1);length=0;}
else if(GKS('X')<0 || GKS('x')<0 || GKS('V')<0 || GKS('v')<0){
  WM_func_CHAR(0);
  if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)<0) close_open(0);
  else close_open(1);length=0;}
else if(GKS('C')<0 || GKS('c')<0){
  WM_func_CHAR(0);
  close_file();length=0;}
else if(GKS('E')<0 || GKS('e')<0 || GKS('T')<0 || GKS('t')<0){
  WM_func_CHAR(0);
  save_all(0);length=0;}
else if(GKS('S')<0 || GKS('s')<0 || GKS('G')<0 || GKS('g')<0){
  WM_func_CHAR(0);
  dlgproc_SAVE(0);length=0;}
else if(GKS('A')<0 || GKS('a')<0){
  WM_func_CHAR(0);
  dlgproc_SAVE(1);length=0;}
else if(GKS('R')<0 || GKS('r')<0){
  WM_func_CHAR(0);
  dlgproc_REN();length=0;}
else if(GKS('I')<0 || GKS('i')<0){
  WM_func_CHAR(0);
  dlgproc_INS();length=0;}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==4*/gotoflag==-1){
  if(GKS(XK_F12)<0 || GKS(XK_Pause)<0 || GKS(XK_F1)<0) cqflag=0;  /* Esc+ */
  BitBltflag=/*2*/1;
  goto end;}

if(gotoflag==1) goto end;else gotoflag=1;
```

```
/*9*/
if(GKS_(ShiftMask)<0 && GKS(XK_F1)<0){
  if(GKS_(Mod1Mask)<0) copy_cfg();
  else use_selector(4);}
else if(GKS(XK_F2)<0){
  if(GKS_(ShiftMask)<0) u_s_flag=1;else u_s_flag=0;
  dlgproc_JUMP();}
else if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)<0 && GKS(XK_Up)<0){
  switch_division(0);}
else if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)<0 && GKS(XK_Down)<0){
  switch_division(1);}
else if(GKS_(ShiftMask)<0 && GKS(XK_F3)<0){
  restore_display();}
else if(GKS(XK_F3)<0){
  divide_display();}
else if((GKS_(ShiftMask)<0 && GKS(XK_F4)<0) || (GKS_(ShiftMask)<0 && GKS(XK_F11)<0)){
  if(GKS_(ControlMask)<0) systemflag=1;else systemflag=0;
  use_selector(3);}
else if(GKS(XK_F4)<0 || GKS(XK_F11)<0){
  show_file();}

else if(GKS_(ControlMask)>=0 && GKS_(ShiftMask)<0 && GKS(XK_F8)<0){
  use_deleted(0);}
else if(GKS_(ControlMask)>=0 && GKS_(ShiftMask)>=0 && GKS(XK_F8)<0){
  use_selector(/*1*/6);}

else if(GKS(XK_Return)<0){
  if(Enter(0)==1) goto end;}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('I')<0 || GKS('i')<0)){
  cqflag=1;prompt_cq(0);}
else if(GKS(XK_Escape)<0){
  cqflag=3;prompt_cq(1);}
else if(GKS(XK_F12)<0){  /* added */
  cqflag=3;prompt_cq(1);cqflag++;}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)<0 && (GKS('Q')<0 || GKS('q')<0)){
  cqflag=5;prompt_cq(2);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('Q')<0 || GKS('q')<0)){
  cqflag=7;prompt_cq(3);}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('Y')<0 || GKS('y')<0)){
  YKP_word(0);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('K')<0 || GKS('k')<0)){
  YKP_word(1);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('M')<0 || GKS('m')<0)){/* added */
```

```
  if(YKP_word(1)==0){
  charflag=0;charcode=2;BitBltflag=/*2*/1;
  Quick_Find(1,1);}}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('N')<0 || GKS('n')<0)){  /* added *
  if(YKP_word(1)==0){
  charflag=0;charcode=2;BitBltflag=/*2*/1;
  Quick_Find(1,2);}}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0
       && (GKS('P')<0 || GKS('p')<0 || GKS('O')<0 || GKS('o')<0)){
  if(GKS('O')<0 || GKS('o')<0) MOVEcsr*=-1;YKP_word(2);
  if(GKS('O')<0 || GKS('o')<0) MOVEcsr*=-1;}
else if(GKS_(ControlMask)<0 && (GKS('Y')<0 || GKS('y')<0)){
  YKP(0);}
else if(GKS_(ControlMask)<0 && (GKS('K')<0 || GKS('k')<0)){
  YKP(1);}
else if(GKS_(ControlMask)<0 &&
       (GKS('P')<0 || GKS('p')<0 || GKS('O')<0 || GKS('o')<0)){
  if(GKS('O')<0 || GKS('o')<0) MOVEcsr*=-1;YKP(2);
  if(GKS('O')<0 || GKS('o')<0) MOVEcsr*=-1;}
else if(GKS_(ControlMask)<0 && GKS('2')<0){
  if(GKS_(Mod1Mask)<0) FILE_jump(1);else FILE_jump(0);}
else if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)<0 && GKS('8')<0){  /* to zzz.string */
  FILE_ref_tmp(1);}
else if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)<0 && GKS('9')<0){  /* to zzz.find */
  FILE_ref_tmp(0);}

else if(GKS(XK_F6)<0){
  paste=0;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS_(ShiftMask)<0 && GKS(XK_F7)<0){
  paste=2;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS(XK_F7)<0){
  paste=1;cut=0;monitorline(1);BitBltflag=2;}

else if(GKS(XK_F5)<0){
  Replace();}
else if(GKS(XK_F9)<0){
  Find(1);}

else {gotoflag=0;}

if(gotoflag==1) goto end;

end_left:
gotoflag=left_keydowns();

if(gotoflag==0 && cqflag>0) goto end_;  /* no job */
```

```
end:
if(length>0){
if(length>1)
WM_funcIME_CHAR(buf_Xmb);                    /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(ftp>0) write_3vals(ftp-1);

if(BitBltflag==0)      {BitBlt_full();csr();}
else if(BitBltflag==1) {monitorline(1);csr();}
else{}

/*no_extraline=0;*/

end_:
return 1;
}/**if(event.type)**/
else if(event.type==Expose){
restore_in_PAINT();

return 1;
}/**else if(event.type)**/
else if(/*event.type==DestroyNotify*/0){
printf(" ?\n");
/*imm_close();*/
XUnsetICFocus(ic);

if(dialogflag>0){
dialogflag=3;refill=0;
breaks(0);
}
else if(menuflag>0){
menuflag=3;refill=0;
breaks(0);
}
else{
close_all();
/*if(BitBltflag==0) {*//*BitBlt_full();*/csr();/*}*/
breaks(1);
}
```

```
return 1;
}/**else if(event.type)**/
else{}

return 0;
}/** wndproc **/


void WM_func_CHAR(unsigned char charcode_tmp)
{
if(charcode_tmp<0x20 || charcode_tmp>0x7e){
if(cqflag>0 && cqflag%2==0){
extraline(1);cqflag=0;
if(filerflag) {if(dialogflag>0 && imm_restart_flag==1) imm_restart();}
else imm_restart();
}
if(cqflag%2==1) cqflag++;

return;
}

if(usflag==1) return;

if(menuflag>0){
if(cqflag==6){
extraline(1);cqflag=0;
if(filerflag) {if(dialogflag>0 && imm_restart_flag==1) imm_restart();}
else imm_restart();
}

return;
}/**if(menuflag)*****************************/

if(dialogflag>0){
charcode=charcode_tmp;

if(cqflag==2){
overwrite();
insertion_cc_dialog(charcode);
extraline(1);cqflag=0;
if(filerflag) {if(imm_restart_flag==1) imm_restart();}
else imm_restart();
}
else if(cqflag==4 || cqflag==6){    /* 4(<- ex. Esc Q) and 6 */
if(noelineflag==0) extraline(1);else noelineflag=0;
```

```
cqflag=0;
if(filerflag) {if(imm_restart_flag==1) imm_restart();}
else imm_restart();
}
else{
overwrite();
insertion_dialog(charcode);
}

return;
}/**if(dialogflag)*****************************/

if(function>=2) return;           /* 3,4 : for Windows ? */
if(filerflag){
if(cqflag==6) {extraline(1);cqflag=0;/*imm_restart();*/goto end;}
return;}

charflag=0;charcode=charcode_tmp;

if(cqflag==2){
overwrite();
insertion_cc(charcode);
extraline(1);cqflag=0;imm_restart();
}
else if(cqflag==4 || cqflag==6 || cqflag==8){
/*if(noelineflag==0) extraline(1);else noelineflag=0;*/  /* <-> save_all() */
cqflag=0;imm_restart();
if(puts_mline_flag) {/*beep(50);*/BitBltflag=2;}   /* <-> save_all() */
}
else{
overwrite();
insertion(charcode);
}

end:{}
}/** WM_func_CHAR **/


void WM_funcIME_CHAR(unsigned char *buf_Xmb)
{
char flag_,function_old,type;
long k,k_sdb;
unsigned char db[2];

strcpy(stock_db,buf_Xmb);
dbsize=strlen(stock_db);
```

```
kmax_sdb=dbsize-1;

if(menuflag>0){
return;
}/**if(menuflag)*********************************/

if(dialogflag>0){
dbflag=1;

k_sdb=0;
while(1){
type=gettype_sdb(k_sdb);

if(type==3){
db[0]=stock_db[k_sdb];
db[1]=stock_db[k_sdb+1];

tailcheck_dialog();

flag_=0;

kmax_dialog+=2;
if(kmax_dialog>ASIZEM-1) {beep(50);kmax_dialog-=2;flag_=1;}
else{
k=firstk_dialog+icsr;
memmove(&p_dialog[k+2],&p_dialog[k],kmax_dialog-2-k+1);
memmove(&p_dialog[k],&db[0],2);
}

/*page_firstk_dialog(firstk_dialog);*/

if(flag_==0){
csr_right_dialog();
}
else{
break;
}

k_sdb+=2;
}/**if(type)**/
else{
if(insertion_dialog(stock_db[k_sdb])==1) break;

k_sdb+=1;
}/**else(type)**/
```

```
if(k_sdb>kmax_sdb) break;
}/**while(1)**/

dbflag=0;
page_firstk_dialog(firstk_dialog);

InputPosition(ic,icsr,jcsr);

return;
}/**if(dialogflag)*****************************/

if(function>=2) return;           /* 3,4 : for Windows ? */
if(filerflag) return;

tailcheck();

k=top_icsr(/*firstline+*/jcsr,icsr);
flag_=pdata_increase(k,&stock_db[0],dbsize);

if(flag_==0 && cut>0 && k<=k_from) k_from+=dbsize;   /* <= */

if(flag_==0){
function_old=function;function=2;
jcsr=while_puts_dline(firstk,k+dbsize);
if(jcsr>ROW-1) {firstk=while_puts_firstk(firstk,jcsr-(ROW-1));jcsr=ROW-1;}
function=function_old;
icsr=icsr_last;
}
page_firstk(firstk);

if(flag_==0){
/*csr_right();monitorline(1);*/
if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;
}

InputPosition(ic,icsr,jcsr);

if(fn!=FMAX-1 && ftp>0) write_3vals(ftp-1);
}/** WM_funcIME_CHAR **/



char left_keydowns(void)
{
char gotoflag,flag_;
long k;
unsigned char db[2];
```

```
gotoflag=1;

if(cqflag==8){                        /* ~^Q-> */
if(GKS('E')<0 || GKS('e')<0){
  switch_division(0);}
else if(GKS('X')<0 || GKS('x')<0){
  switch_division(1);}
else if(GKS('V')<0 || GKS('v')<0){
  move_and_paste();}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

/*9*/
if(/*cqflag==8*/gotoflag==-1){
  if(GKS(XK_F12)<0 || GKS(XK_Pause)<0 || GKS(XK_F1)<0) cqflag=0;  /* Esc+ */
  BitBltflag=/*2*/1;
  goto end_lk;}

if(gotoflag==1) goto end_lk;else gotoflag=1;

if(cqflag==6){                        /* ^Q-> */
if(GKS('7')<0 && cut==0 && filerflag==0){
  hcentering(0);}
else if(GKS('8')<0 && cut==0 && filerflag==0){
  hcentering(1);}
else if(GKS('9')<0 && cut==0 && filerflag==0){
  hcentering(2);}

else if((GKS('M')<0 || GKS('m')<0) && cut==0 && filerflag==0 &&
        refflag==0 && deletedflag==0){
  file_attri();}

else if((GKS('H')<0 || GKS('h')<0) && cut==0 && filerflag==0){
  half_word(0);}
else if((GKS('G')<0 || GKS('g')<0) && cut==0 && filerflag==0){
  half_word(1);}
else if((GKS('Y')<0 || GKS('y')<0) && cut==0 && filerflag==0){
  half_line(0);}
else if((GKS('T')<0 || GKS('t')<0) && cut==0 && filerflag==0){
  half_line(1);}

else if(GKS('E')<0 || GKS('e')<0){
  csr_column_home();}
```

```
else if(GKS('X')<0 || GKS('x')<0){
  csr_column_end();}
else if((GKS('S')<0 || GKS('s')<0) && filerflag==0){
  csr_row_home();}
else if((GKS('D')<0 || GKS('d')<0) && filerflag==0){
  csr_row_end();}

else if((GKS('F')<0 || GKS('f')<0) && filerflag==0){
  find_0x1a(0);}
else if((GKS('A')<0 || GKS('a')<0) && filerflag==0){
  find_0x1a(1);}

else if(GKS('R')<0 || GKS('r')<0){
  text_home();}
else if(GKS('C')<0 || GKS('c')<0){
  text_end();}

else if(GKS('W')<0 || GKS('w')<0){
  page_up();}
else if(GKS('Z')<0 || GKS('z')<0){
  page_down();}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==6*/gotoflag==-1){
  if(GKS(XK_F12)<0 || GKS(XK_Pause)<0 || GKS(XK_F1)<0) cqflag=0;  /* Esc+ */
  BitBltflag=/*2*/1;
  goto end_lk;}

if(gotoflag==1) goto end_lk;else gotoflag=1;

if(GKS_(ControlMask)<0 && (GKS('B')<0 || GKS('b')<0) && filerflag==0){
  cut=1;BL();}
else if(GKS_(ControlMask)<0 && (GKS('L')<0 || GKS('l')<0) && filerflag==0){
  cut=2;BL();}

else if(GKS_(ShiftMask)<0 && GKS(XK_Delete)<0 && filerflag==0){
  half_word(0);}
else if(GKS_(ShiftMask)<0 && GKS(XK_BackSpace)<0 && filerflag==0){
#if NOTEKBD==0
  half_word(1);
#else
  delorbs=0;
  deletion();
```

```
#endif
  }
else if(GKS_(ControlMask)<0 && GKS(XK_Delete)<0 && filerflag==0){
  half_line(0);}
else if(GKS_(ControlMask)<0 && GKS(XK_BackSpace)<0 && filerflag==0){
  half_line(1);}

else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_Delete)<0 &&
        filerflag==0){
  delorbs=0;
  deletion();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_BackSpace)<0 &&
        filerflag==0){
  delorbs=1;
  backspace();}
else if(GKS_(ControlMask)<0 && (GKS('U')<0 || GKS('u')<0) && filerflag==0){
  overwrite();
  insertion_u();}

else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)<0 && GKS_(Mod1Mask)>=0 &&
        GKS(XK_Up)<0){
  csr_column_home();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)<0 && GKS_(Mod1Mask)>=0 &&
        GKS(XK_Down)<0){
  csr_column_end();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_Home)<0 &&
        filerflag==0){
  csr_row_home();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_End)<0 &&
        filerflag==0){
  csr_row_end();}

else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_Up)<0){
  csr_up();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_Down)<0){
  csr_down();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_Left)<0 &&
        filerflag==0){
  csr_left();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_Right)<0 &&
        filerflag==0){
  csr_right();}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)>=0 && GKS(XK_Prior)<0){
  scroll_up(0);}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)>=0 && GKS(XK_Next)<0){
```

```
  scroll_down(0);}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)<0 && GKS(XK_Home)<0){
  centering_csr();}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)>=0 && GKS(XK_Home)<0){
  centering_theline();}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && GKS(XK_Home)<0){
  page_firstk(topp[jcsr]);
  jcsr=0;/*csr();*/
  }

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('T')<0 || GKS('t')<0)){
  page_firstk(topp[jcsr]);
  jcsr=0;/*csr();*/
  }
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('V')<0 || GKS('v')<0)){
  centering_theline();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)<0 && (GKS('V')<0 || GKS('v')<0)){
  centering_csr();}

else if((GKS_(ControlMask)<0 && GKS(XK_Prior)<0) ||
        (GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && GKS(XK_Up)<0)){
  text_home();}
else if((GKS_(ControlMask)<0 && GKS(XK_Next)<0) ||
        (GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && GKS(XK_Down)<0)){
  text_end();}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)>=0 && GKS(XK_Up)<0){
  page_up();}
else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)>=0 && GKS(XK_Down)<0){
  page_down();}

else if(GKS(XK_Prior)<0){
  if(function==0) page_up();
  else{
  charflag=0;charcode=0;
  BitBltflag=2;}}
else if(GKS(XK_Next)<0){
  if(function==0) page_down();
  else{
  charflag=0;charcode=1;
  BitBltflag=2;}}

else if(GKS_(ControlMask)<0 && GKS(XK_Right)<0 && filerflag==0){
  find_0x1a(0);}
else if(GKS_(ControlMask)<0 && GKS(XK_Left)<0 && filerflag==0){
  find_0x1a(1);}
```

```
else if(GKS_(ShiftMask)<0 && GKS(XK_Right)<0 && filerflag==0){
  find_word(0);}
else if(GKS_(ShiftMask)<0 && GKS(XK_Left)<0 && filerflag==0){
  find_word(1);}

else if(GKS_(ShiftMask)<0 && GKS(XK_Insert)<0 && refflag==0){
  charflag=0;charcode=2;
  BitBltflag=2;}
else if((GKS(XK_Pause)<0 || (GKS_(ShiftMask)>=0 && GKS(XK_F1)<0)) && refflag==0){/* added */
  charflag=0;charcode=2;
  BitBltflag=2;}
else if(GKS_(ControlMask)<0 && GKS(XK_Insert)<0 && filerflag==0){
  if(insorover==0) insorover=1;else insorover=0;
  extraline(1);BitBltflag=2;}
else if(GKS(XK_Tab)<0 && filerflag==0){
  overwrite();
  insertion(0x09);}

else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)<0 && (GKS('G')<0 || GKS('g')<0) &&
        filerflag==0){
  delorbs=0;
  deletion();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)<0 && (GKS('H')<0 || GKS('h')<0) &&
        filerflag==0){
  delorbs=1;
  backspace();}

else if(GKS_(ControlMask)<0 && (GKS('E')<0 || GKS('e')<0)){
  csr_up();}
else if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)>=0 && (GKS('X')<0 || GKS('x')<0)){
  csr_down();}
else if(GKS_(ControlMask)<0 && (GKS('S')<0 || GKS('s')<0) && filerflag==0){
  csr_left();}
else if(GKS_(ControlMask)<0 && (GKS('D')<0 || GKS('d')<0) && filerflag==0){
  csr_right();}

else if(GKS_(ControlMask)<0 && (GKS('F')<0 || GKS('f')<0) && filerflag==0){
  find_word(0);}
else if(GKS_(ControlMask)<0 && (GKS('A')<0 || GKS('a')<0) && filerflag==0){
  find_word(1);}

else if(GKS_(ControlMask)<0 && (GKS('R')<0 || GKS('r')<0)){
  if(function==0) page_up();
  else{
  charflag=0;charcode=0;
  BitBltflag=2;}}
```

```
else if(GKS_(ControlMask)<0 && (GKS('C')<0 || GKS('c')<0)){
  if(function==0) page_down();
  else{
  charflag=0;charcode=1;
  BitBltflag=2;}}

else if(GKS_(ControlMask)<0 && (GKS('9')<0) && refflag==0){
  Quick_Find(0,0);}

else if(GKS_(ControlMask)<0 && GKS_(Mod1Mask)>=0 && (GKS('W')<0 || GKS('w')<0)){
  scroll_up(0);}
else if(GKS_(ControlMask)<0 && (GKS('Z')<0 || GKS('z')<0)){
  scroll_down(0);}

else if(GKS_(ControlMask)<0 && (GKS('M')<0 || GKS('m')<0)){
  if(refflag){
  if(GKS_(ShiftMask)<0/* || GKS_(ControlMask)<0*/){
  /*if(GKS_(ShiftMask)<0) {*/uflag=1;csr_row_home();/*}*/

  if(insertion('\n')==1) {uflag=0;goto end_lk;}

  /*if(GKS_(ShiftMask)<0) */uflag=0;}
  else{
  refill=-2;charflag=0;charcode=2;
  BitBltflag=2;}}/**if(refflag)**/
  else if(filerflag){
  refill=-2;if(GKS_(ShiftMask)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}/**else if(filerflag)**/
  else{
  if(GKS_(ShiftMask)<0) {uflag=1;csr_row_home();}
  if(AINDENT==1) /*{if(GKS_(ControlMask)>=0) */lumpflag=1;/*}*/
  /*else {if(GKS_(ControlMask)<0) lumpflag=1;}*/

  if(insertion('\n')==1) {lumpflag=0;uflag=0;goto end_lk;}

  if(AINDENT==1) /*{if(GKS_(ControlMask)>=0) */autoindent();/*}*/
  /*else {if(GKS_(ControlMask)<0) autoindent();}*/
  if(GKS_(ShiftMask)<0) uflag=0;}/**else(refflag,filerflag)**/}
else if(GKS_(ControlMask)<0 && (GKS('N')<0 || GKS('n')<0) && filerflag==0){
  if(refflag){
  /*if(GKS_(ShiftMask)<0) {*/uflag=1;csr_row_home();/*}*/

  if(insertion('\n')==1) {uflag=0;goto end_lk;}

  /*if(GKS_(ShiftMask)<0) */uflag=0;}/**if(refflag)**/
  else{
```

```
  /*if(GKS_(ShiftMask)<0) {*/uflag=1;csr_row_home();/*}*/
  if(AINDENT==1) /*{if(GKS_(ControlMask)>=0) */lumpflag=1;/*}*/
  /*else {if(GKS_(ControlMask)<0) lumpflag=1;}*/

  if(insertion('\n')==1) {lumpflag=0;uflag=0;goto end_lk;}

  if(AINDENT==1) /*{if(GKS_(ControlMask)>=0) */autoindent();/*}*/
  /*else {if(GKS_(ControlMask)<0) autoindent();}*/
  /*if(GKS_(ShiftMask)<0) */uflag=0;}/**else(refflag)**/}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 &&
        (GKS('H')<0 || GKS('h')<0) && filerflag==0){
  lumpflag=1;

  uflag=1;
  if(insertion(' ')==1) {lumpflag=0;uflag=0;goto end_lk;}
  uflag=0;

  csr_left();
  lumpflag=0;
  page_firstk(firstk);}

else{
BitBltflag=2;if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}
gotoflag=0;
}

end_lk:{}
return gotoflag;
}/** left_keydowns **/


void left_keydowns_dialog(void)
{
char gotoflag;

gotoflag=1;

if(cqflag==6){
if(GKS('S')<0 || GKS('s')<0){
  csr_row_home_dialog();}
else if(GKS('D')<0 || GKS('d')<0){
  csr_row_end_dialog();}

else {gotoflag=-1;}
}/**if(cqflag)**/
```

```
else {gotoflag=0;}

if(/*cqflag==6*/gotoflag==-1){
  if(GKS(XK_F12)<0 || GKS(XK_Pause)<0 || GKS(XK_F1)<0) cqflag=0;  /* Esc+ */
  /*BitBltflag_=2;*/
  goto end_lk_dialog;}

if(gotoflag==1) goto end_lk_dialog;else gotoflag=1;

if(GKS(XK_Delete)<0){
  deletion_dialog();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_BackSpace)<0){
  backspace_dialog();}

else if(GKS(XK_Up)<0){
  restore_dialog();}
else if(GKS(XK_Down)<0){
  clear_dialog(1);}
else if(GKS_(ControlMask)>=0 && GKS(XK_Left)<0){
  csr_left_dialog();}
else if(GKS_(ControlMask)>=0 && GKS(XK_Right)<0){
  csr_right_dialog();}

else if(GKS(XK_Home)<0 || (GKS_(ControlMask)<0 && GKS(XK_Left)<0)){
  csr_row_home_dialog();}
else if(GKS(XK_End)<0 || (GKS_(ControlMask)<0 && GKS(XK_Right)<0)){
  csr_row_end_dialog();}

else if(GKS(XK_Prior)<0){
  page_up_dialog();}
else if(GKS(XK_Next)<0){
  page_down_dialog();}

else if(GKS_(ControlMask)<0 && GKS(XK_Insert)<0){
  if(insorover==0) insorover=1;else insorover=0;
  extraline(1);BitBltflag_=2;}

else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('I')<0 || GKS('i')<0)){
  cqflag=1;prompt_cq(0);}
else if(GKS_(ControlMask)<0 && (GKS('Q')<0 || GKS('q')<0)){
  cqflag=5;prompt_cq(2);}
else if(GKS(XK_Tab)<0){
  overwrite();
  insertion_dialog(0x09);}

else if(GKS_(ControlMask)<0 && (GKS('G')<0 || GKS('g')<0)){
```

```
  deletion_dialog();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)<0 && (GKS('H')<0 || GKS('h')<0)){
  backspace_dialog();}

else if(GKS_(ControlMask)<0 && (GKS('E')<0 || GKS('e')<0)){
  restore_dialog();}
else if(GKS_(ControlMask)<0 && (GKS('X')<0 || GKS('x')<0)){
  clear_dialog(1);}
else if(GKS_(ControlMask)<0 && (GKS('S')<0 || GKS('s')<0)){
  csr_left_dialog();}
else if(GKS_(ControlMask)<0 && (GKS('D')<0 || GKS('d')<0)){
  csr_right_dialog();}

else if(GKS_(ControlMask)<0 && (GKS('R')<0 || GKS('r')<0)){
  page_up_dialog();}
else if(GKS_(ControlMask)<0 && (GKS('C')<0 || GKS('c')<0)){
  page_down_dialog();}

else if(GKS_(ControlMask)<0 && (GKS('M')<0 || GKS('m')<0)){
  trim_dialog();
  if(GKS_(/*ControlMask*/ShiftMask)<0) use_selector_flag=1;else use_selector_flag=0;
  dialogflag=2;refill=0;BitBltflag_=2;}

else {BitBltflag_=2;}

end_lk_dialog:{}
}/** left_keydowns_dialog **/


void left_keydowns_menu(void)
{
char gotoflag;

gotoflag=1;

if(cqflag==6){
if(GKS('E')<0 || GKS('e')<0){
  csr_column_home_menu();}
else if(GKS('X')<0 || GKS('x')<0){
  csr_column_end_menu();}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==6*/gotoflag==-1){
```

```
  if(GKS(XK_F12)<0 || GKS(XK_Pause)<0 || GKS(XK_F1)<0) cqflag=0;  /* Esc+ */
  /*BitBltflag_=2;*/
  goto end_lk_menu;}

if(gotoflag==1) goto end_lk_menu;else gotoflag=1;

if(GKS_(ControlMask)<0 && GKS(XK_Up)<0){
  csr_column_home_menu();}
else if(GKS_(ControlMask)<0 && GKS(XK_Down)<0){
  csr_column_end_menu();}

else if(GKS(XK_Up)<0){
  csr_up_menu();}
else if(GKS(XK_Down)<0){
  csr_down_menu();}

else if(GKS_(ControlMask)<0 && (GKS('Q')<0 || GKS('q')<0)){
  cqflag=5;prompt_cq(2);}

else if(GKS_(ControlMask)<0 && (GKS('E')<0 || GKS('e')<0)){
  csr_up_menu();}
else if(GKS_(ControlMask)<0 && (GKS('X')<0 || GKS('x')<0)){
  csr_down_menu();}

else if(GKS_(ControlMask)<0 && (GKS('M')<0 || GKS('m')<0)){
  menuflag=2;refill=0;BitBltflag_=2;}

else {BitBltflag_=2;}

end_lk_menu:{}
}/** left_keydowns_menu **/
```