Fine soccer(4)

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. Processing of score

In the 2nd, it is assumed that we get processing score from experimental score in E league. In this league, If CK, DFK occur, before them, we do LK, MK, NK of which runs are less than usual goal and we get the runs. Total runs namely experimental score is fixed by the number of usual goals and these runs. For the sake of ease, NK due to DFK is omitted. It is assumed that the runs of usual goal is 7 and the runs of goal by LK and the runs of goal by MK are 3 and 1 respectively. Experimental score is "the number of usual goals" $\times 7 +$"the number of goals by LK" $\times 3 +$"the number of goals by MK" $\times 1$.

For example, we assume that team 1 and team 2 end a game 1 to 1. The number of CKs is 5 on both and we assume that the following runs are got by LK, MK.

Table 1

CK	：1st	2nd	3rd	4th	5th
run(s)(Team 1)	：(-7)+3	0	1	1	0
run(s)(Team 2)	：(-7)+3	0	1	1	0

The runs 3 is runs of goal by LK. (-7) is the term in the case that after CK, an usual goal is got the ball not going over center line, the ball not being catched by GK and goal kick not being done. Namely, if LK is chosen in 1st CK, the usual goal is cancelled. If the ball goes over center line, the ball is catched by GK or goal kick is done after 1st CK, the 1st becomes (0)+3.

In the above table, if all the runs on CK is added, experimental score becomes as follows:

```
run(s)(Team 1)：7+(-7)+3+0+1+1+0
run(s)(Team 2)：7+(-7)+3+0+1+1+0
```

In the above scores, not CK but LK, MK are chosen in all CKs. In an actual game, such choice is not always done. So, we try varying the choice number $n_i (i = 1, 2)$ of LKs, MKs. If a program is used, the combination patterns are got easily.

```
int num1;
int i1,j1,k1,l1,m1;

num1=2;

for(i1=0;i1<2;i1++)
for(j1=0;j1<2;j1++)
for(k1=0;k1<2;k1++)
for(l1=0;l1<2;l1++)
for(m1=0;m1<2;m1++){
if(i1+j1+k1+l1+m1==num1)
```

```
printf(" %d %d %d %d %d\n",i1,j1,k1,l1,m1);
}
```

In the above program, the choice number n_1(num1) of LKs, MKs on team 1 is 2 and the output is as following. 0 and 1 represent nonchoice and choice respectively.

```
 0 0 0 1 1     /* 1st 2nd 3rd 4th 5th */
 0 0 1 0 1
 0 0 1 1 0
 0 1 0 0 1
 0 1 0 1 0
 0 1 1 0 0
 1 0 0 0 1
 1 0 0 1 0
 1 0 1 0 0
 1 1 0 0 0
```

We get scores which use the runs of the previous table namely precessing scores and we get wins, draws, losses and the probability to win on team 1. Because the choice number n_2(num2) of LKs, MKs on team 2 is assumed to be 0, the processing score on team 2 is 7. The above output lines correspond to the following runs and results.

```
processing score on team 1=7+1+0=8>7 : win
processing score on team 1=7+1+0=8>7 : win
processing score on team 1=7+1+1=9>7 : win
processing score on team 1=7+0+0=7=7 : draw
processing score on team 1=7+0+1=8>7 : win
processing score on team 1=7+0+1=8>7 : win
processing score on team 1=7+(-7)+3+0=3<7 : loss
processing score on team 1=7+(-7)+3+1=4<7 : loss
processing score on team 1=7+(-7)+3+1=4<7 : loss
processing score on team 1=7+(-7)+3+0=3<7 : loss
```

The program is as follows:

```
int num1,num2;
int i1,j1,k1,l1,m1,R1_ini,R1;
int i2,j2,k2,l2,m2,R2_ini,R2;
int win,draw,loss,P;
int rck1[5][2],rck2[5][2];          /* 5:i?, j?, k?, l?, m? */
double P;

num1=2;
num2=0;

R1_ini=7;
R2_ini=7;
```

```
for(i1=0;i1<5;i1++){
rck1[i1][0]=0;
rck2[i1][0]=0;
}

rck1[0][1]=(-7)+3;  /* (-7)+3 or (0)+3 */
rck1[1][1]=0;
rck1[2][1]=1;
rck1[3][1]=1;
rck1[4][1]=0;

rck2[0][1]=(-7)+3;  /* (-7)+3 or (0)+3 */
rck2[1][1]=0;
rck2[2][1]=1;
rck2[3][1]=1;
rck2[4][1]=0;

win=0;
draw=0;
loss=0;

for(i1=0;i1<2;i1++)
for(j1=0;j1<2;j1++)
for(k1=0;k1<2;k1++)
for(l1=0;l1<2;l1++)
for(m1=0;m1<2;m1++){
if(i1+j1+k1+l1+m1==num1){
R1=R1_ini+rck1[0][i1]+rck1[1][j1]+rck1[2][k1]+rck1[3][l1]+rck1[4][m1];

for(i2=0;i2<2;i2++)
for(j2=0;j2<2;j2++)
for(k2=0;k2<2;k2++)
for(l2=0;l2<2;l2++)
for(m2=0;m2<2;m2++){
if(i2+j2+k2+l2+m2==num2){
R2=R2_ini+rck2[0][i2]+rck2[1][j2]+rck2[2][k2]+rck2[3][l2]+rck2[4][m2];

if(R1>R2) win++;
else if(R1==R2) draw++;
else loss++;
}/**if(i2,j2,...)**/
}/**for(m2)**/

}/**if(i1,j1,...)**/
}/**for(m1)**/
```

In the above program, n_1(num1) on team 1 is 2 and n_2(num2) on team 2 is 0, so the probability P to

win on team 1 is as follows:

```
printf(" win:%d draw:%d loss:%d\n",win,draw,loss);  /* win:5 draw:1 loss:4 */
P=(win+draw*0.5)/(win+draw+loss);
printf(" P:%f\n",P);                  /* P:0.55 */
```

On team 1, we get the probability to win from all the combinations choosing two CKs out of five CKs. Therefore, it is a probability in the case that in enough times games, choosing two CKs under equal probabilities, LK, MK are substituted for them. Namely, it is an imajinary and average probability to win on games of Table 1.

If n_1, n_2 are varied from 0 to 5 independently, the probability varies like Figure 1. The figures in the circle is "the probability" $\times$ 100 and the gradation of color reflects it.

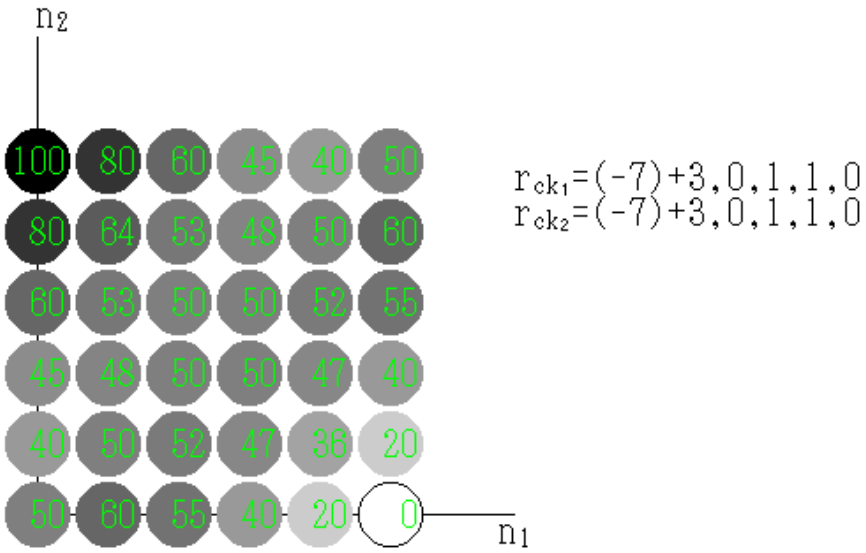


Figure 1

2. Signal

Figure 2 shows examples of sign with two arms : 'T' or 'O' of acceptance signal which referee sends out after call of time-out.

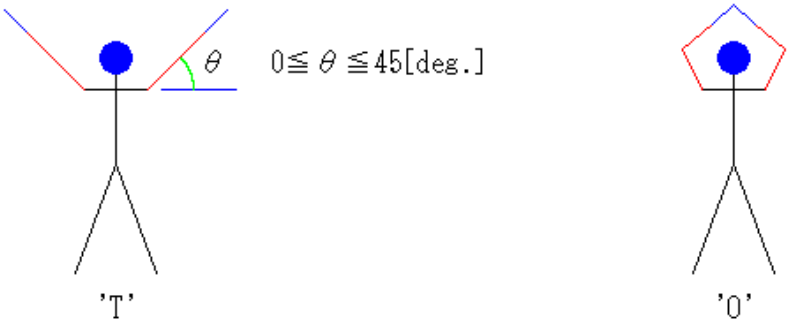
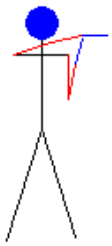
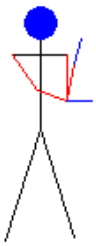


Figure 2

Figure 3 shows examples of contact type of sign with two arms : 'T'.



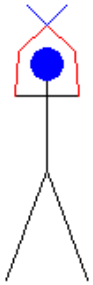
side 'T'



inverse 'T'

Figure 3

In (B) in "8. Transmission of call" in the 2nd, time-out begins not waiting for signal of referee. If call of time-out is transmitted to referee outside the range of gap or break, referee sends out rejection signal. Figure 4 shows an example of sign with two arms : 'X' of rejection signal.



'X'

Figure 4

3. Whistle

If short whistles are added when referee sends out a signal, the reception of it becomes sure. The number of times of short whistles is as follows:

- rejection of time-out(sign with two arms : 'X') : 3
- acceptance of time-out(sign with two arms : 'T' or 'O') : 2

Because the number of times of short horns of call of time-out is 3, the number of times of short horns of rejection of time-out is 3 which is the same as it and the number of times of short horns of acceptance of time-out is 2 which is smaller by 1 than it. If call of time-out is done when a ball or players are active, time-out is rejected. Therefore, referee must whistle 3 times clearly in order for all players to be able to recognize rejection of time-out and activeness of a ball or players. In addition to it, referee make the direction of referee's body right in order for players who are near a ball and players who are in the direction of movement of the ball to be able to recognize sign with two arms : 'X' visually.

4. Free kick

In 7 persons rugby, we resume a game with scrum after knock on, throw forward like 15 persons rugby. However, in 7 persons rugby, it seems that a ball is put out from scrum right away and the position of scrum haldly changes. It seems that scrum is only a formality. So, we try considering a resumption way of game which suits 7 persons rugby. Because knock on, throw forward are slight fouls, we recommend that a free kick of which advantage is smaller than penalty kick can be chosen. If

precision of kick is high, speed of kicked ball is high, sign play is minute, player can catch a ball skillfully and player can run fast, player can approach goal line in proportion to them. We make such situation. In Figure 5, $\otimes$ is scrumming point and kicker kicks a ball at this point if not scrum but free kick is chosen. Players except kicker can not be before the separation line in the figure before the kick. The distance Δx from scrumming point of the separation line is 10m for example.

We often see a scene that kicker catches a ball which kicker kicked. In this free kick, there two ways of catching the ball.

- catch of kicker is accepted
- catch of kicker is not accepted

Because kicker is marked too in the former, the possibility of getting runs in the former becomes a little higher than the latter.

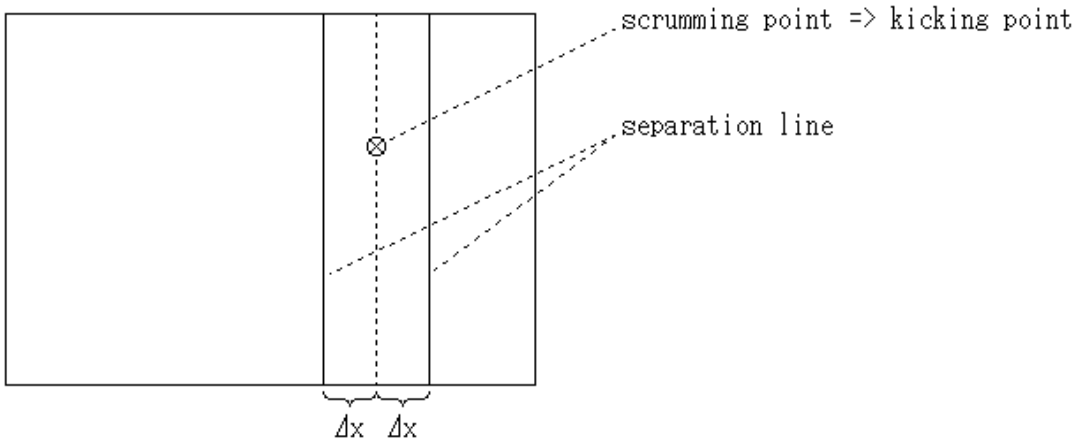


Figure 5

5. Time restriction in maul

There is 5sec rule on termination of maul in maul. We try considering a new time restriction rule which can exist on a parallel with this rule. We assume that a player must pass a ball if a time Δt goes by after formation of maul. A player to whom a ball is passed is a player except maul members in Δt. If mean weight is heavy, it becomes an advantage in maul like scrum. If the rule is adopted, because if mean weight is light, it will be directly connected with high mobility, the disadvantage of light mean weight will be made up to some extent utilizing high mobility. Δt is assumed to be 8sec for example.

****

ファインサッカー (4)

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. スコアの加工

第二回ではEリーグにおいて実験スコアから加工スコアを求めるとしました。このリーグではCK、DFKが発生したらその前に得点が通常のゴールよりも小さいLK、MK、NKを行い、これらによる得点を求めます。総得点すなわち実験スコアは通常のゴールの数とこれらの得点で決まります。簡単のため、DFK起因のNKは省略します。通常のゴールの得点は7、LK、MKによるゴールの得点は各々3、1とします。実験スコアは、通常のゴール数×7＋LKによるゴールの数×3＋MKによるゴールの数×1となります。

たとえば、チーム1とチーム2が1対1で試合を終えたとします。CKの数は共に5で、LK、MKによって以下のように得点されたとします。

表 1

CK	：1st	2nd	3rd	4th	5th
run(s)(Team 1)	：(-7)+3	0	1	1	0
run(s)(Team 2)	：(-7)+3	0	1	1	0

得点3はLKによるゴールの得点です。(-7)はCKの後で、ボールがセンターラインを越えず、かつボールがGKにキャッチされず、かつゴールキックがなされずに通常のゴールが決まった場合の項です。つまり、1st CKにおいてCKではなくLKを選択すれば、その通常のゴールはなかったことにする訳です。もし、1st CKの後で、ボールがセンターラインを越えるか、またはボールがGKにキャッチされるか、またはゴールキックがなされれば1stは(0)+3となります。

上の表においてすべてのCKの得点を加えると実験スコアは以下のようになります。

```
run(s)(Team 1)：7+(-7)+3+0+1+1+0
run(s)(Team 2)：7+(-7)+3+0+1+1+0
```

これはすべてのCKにおいてCKではなくLK、MKを選択した場合です。実際の試合では必ずしもこのような選択をするとは限りません。そこで、LK、MKの選択の数$n_i(i=1,2)$を変化させてみます。プログラムを用いると組合せのパターンは容易に求められます。

```
int num1;
int i1,j1,k1,l1,m1;

num1=2;

for(i1=0;i1<2;i1++)
for(j1=0;j1<2;j1++)
for(k1=0;k1<2;k1++)
```

```
for(l1=0;l1<2;l1++)
for(m1=0;m1<2;m1++){
if(i1+j1+k1+l1+m1==num1)
printf(" %d %d %d %d %d\n",i1,j1,k1,l1,m1);
}
```

上のプログラムではチーム 1 の LK、MK の選択の数 $n_1 = 2$(num1=2) であり、その出力は以下のようになります。0,1 が各々非選択, 選択を表しています。

```
 0 0 0 1 1     /* 1st 2nd 3rd 4th 5th */
 0 0 1 0 1
 0 0 1 1 0
 0 1 0 0 1
 0 1 0 1 0
 0 1 1 0 0
 1 0 0 0 1
 1 0 0 1 0
 1 0 1 0 0
 1 1 0 0 0
```

先の表の得点を用いたスコアすなわち加工スコアを求め、チーム 1 の勝ち、引分け、負けおよび勝つ確率を求めます。チーム 2 の LK、MK の選択の数は $n_2 = 0$ とするので、チーム 2 の加工スコア=7 となります。上の出力行は以下のような得点と結果になります。

チーム 1 の加工スコア=7+1+0=8>7：勝ち
チーム 1 の加工スコア=7+1+0=8>7：勝ち
チーム 1 の加工スコア=7+1+1=9>7：勝ち
チーム 1 の加工スコア=7+0+0=7=7：引分け
チーム 1 の加工スコア=7+0+1=8>7：勝ち
チーム 1 の加工スコア=7+0+1=8>7：勝ち
チーム 1 の加工スコア=7+(-7)+3+0=3<7：負け
チーム 1 の加工スコア=7+(-7)+3+1=4<7：負け
チーム 1 の加工スコア=7+(-7)+3+1=4<7：負け
チーム 1 の加工スコア=7+(-7)+3+0=3<7：負け

プログラムは以下のようになります。

```
int num1,num2;
int i1,j1,k1,l1,m1,R1_ini,R1;
int i2,j2,k2,l2,m2,R2_ini,R2;
int win,draw,loss;
int rck1[5][2],rck2[5][2];          /* 5:i?, j?, k?, l?, m? */
double P;

num1=2;
num2=0;

R1_ini=7;
```

```
R2_ini=7;

for(i1=0;i1<5;i1++){
rck1[i1][0]=0;
rck2[i1][0]=0;
}

rck1[0][1]=(-7)+3;  /* (-7)+3 or (0)+3 */
rck1[1][1]=0;
rck1[2][1]=1;
rck1[3][1]=1;
rck1[4][1]=0;

rck2[0][1]=(-7)+3;  /* (-7)+3 or (0)+3 */
rck2[1][1]=0;
rck2[2][1]=1;
rck2[3][1]=1;
rck2[4][1]=0;

win=0;
draw=0;
loss=0;

for(i1=0;i1<2;i1++)
for(j1=0;j1<2;j1++)
for(k1=0;k1<2;k1++)
for(l1=0;l1<2;l1++)
for(m1=0;m1<2;m1++){
if(i1+j1+k1+l1+m1==num1){
R1=R1_ini+rck1[0][i1]+rck1[1][j1]+rck1[2][k1]+rck1[3][l1]+rck1[4][m1];

for(i2=0;i2<2;i2++)
for(j2=0;j2<2;j2++)
for(k2=0;k2<2;k2++)
for(l2=0;l2<2;l2++)
for(m2=0;m2<2;m2++){
if(i2+j2+k2+l2+m2==num2){
R2=R2_ini+rck2[0][i2]+rck2[1][j2]+rck2[2][k2]+rck2[3][l2]+rck2[4][m2];

if(R1>R2) win++;
else if(R1==R2) draw++;
else loss++;
}/**if(i2,j2,...)**/
}/**for(m2)**/

}/**if(i1,j1,...)**/
```

```
}/**for(m1)**/
```

上のプログラムではチーム 1 は $n_1 = 2$(num1=2)、チーム 2 は $n_2 = 0$(num2=0) であり、チーム 1 の勝つ確率 P は以下のようになります。

```
printf(" win:%d draw:%d loss:%d\n",win,draw,loss);  /* win:5 draw:1 loss:4 */
P=(win+draw*0.5)/(win+draw+loss);
printf(" P:%f\n",P);                    /* P:0.55 */
```

チーム 1 に関しては、五つの CK から二つの CK を選んでそのすべての組合せから勝つ確率 P を求めています。したがって、この確率は、十分大きな回数の試合において等確率で二つの CK を選んでそれらを LK、MK に置き換えた場合の確率です。つまり、表 1 の試合の仮想的で平均的な勝つ確率です。

n_1、n_2 を各々0 から 5 まで変化させると確率は図 1 のように変化します。円内の数字は確率× 100 であり、色の濃淡は確率を反映しています。

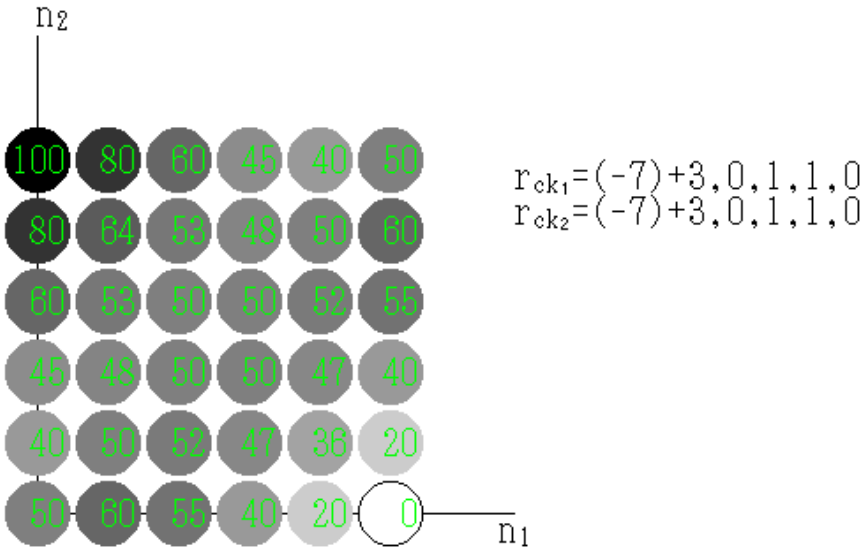


図 1

2. シグナル

タイムアウトがコールされた後にレフェリーが発する容認のシグナルの腕文字'T' または'O' の例を図 2 に示します。

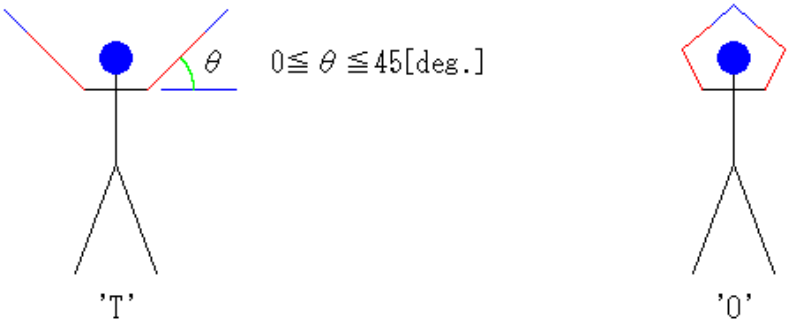


図 2

腕文字'T' の接触型の例を図 3 に示します。

図 3

前回の”8. コールの伝達”における (B) ではレフェリーのシグナルを待たずにタイムアウトが始まります。もしタイムアウトのコールがギャップまたはブレイクの範囲外にレフェリーに伝われればレフェリーは却下のシグナルを発します。却下のシグナルの腕文字’X’ の例を図 4 に示します。

図 4

3. ホイッスル

レフェリーがシグナルを発信する際に短いホイッスルを添えるとシグナルの受信が確実になります。短いホイッスルの回数は以下の通りです。

・タイムアウトの却下 (腕文字：’X’)　　：3
・タイムアウトの容認 (腕文字：’T’ または’O’)：2

タイムアウトのコールのショートホーンの回数は 3 ですから、タイムアウトの却下の短いホイッスルの回数はそれと同じ 3 であり、タイムアウトの容認の短いホイッスルの回数はそれより一つ少ない 2 です。ボール、プレーヤーがアクティブである場合にタイムアウトのコールがなされればタイムアウトは却下されます。したがって、レフェリーは、すべてのプレーヤーがタイムアウトの却下とボール、プレーヤーがアクティブであることを認識できるよう、明確に 3 回ホイッスルしなければなりません。その際、レフェリーは、ボールの近くにいるプレーヤーとボールの進行方向にいるプレーヤーが腕文字’X’ を視覚的に認識できるよう、体の向きを適正にします。

4. フリーキック

7 人制ラグビーでは 15 人制ラグビーと同様、ノックオン、スローフォワードの後はスクラムで試合を再開します。ただし、7 人制ではボールはすぐにスクラムの外に出され、スクラムの位置はほとんど変化しないようです。スクラムは形式的なものになってしまっているようです。そこで、7 人制の特徴に合った試合の再開方法を考えてみます。ノックオン、スローフォワードは軽微な反則ですから、ペナルティキックよりはアドバンテージが小さいフリーキックを選択できるようにしたらどうでしょう。キッ

クの精度が高く、キックされたボールが速く、サインプレーが緻密で、プレーヤーがボールを上手にキャッチでき、速く走ることができれば、ゴールラインにそれだけ近づくことができるようなシチュエーションを作りましょう。図 5 において、⊗ はスクラム点であり、スクラムではなくフリーキックを選択すればキッカーはこの点でボールをキックします。キッカー以外のプレーヤーはキックする前は図の離隔線より前に出てはいけません。離隔線のスクラム点からの距離 Δx はたとえば 10m とします。

よくキッカーがキックされたボールを自身がキャッチしてパスするシーンを見ます。このフリーキックではキャッチに関して二通りを想定しています。

・キッカーのキャッチを認める
・キッカーのキャッチを認めない

前者ではキッカーもマークされるので、前者の得点の可能性が後者より少し高くなります。

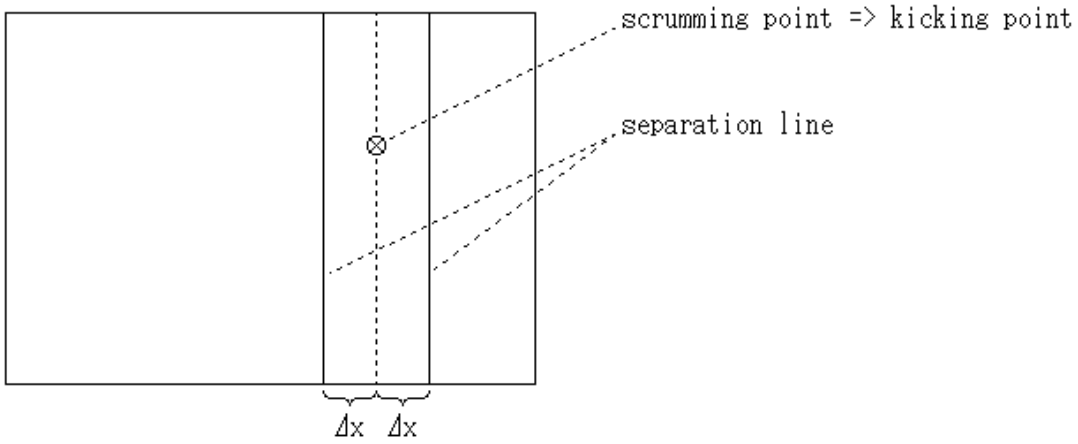


図 5

5. モールの時間制限

モールにはモールの終了に関する 5 秒ルールがあります。このルールと並存できる新たな時間制限ルールを考察してみます。モールが形成されてからある時間 Δt 経てばボールをパスしなければならない、としましょう。ボールをパスされるプレーヤーは Δt 経過時のモール構成員以外のプレーヤーです。モールはスクラムと同様平均体重が重い方が有利です。このルールを採用すれば、平均体重が軽い方が機動性が高いでしょうから、機動性の高さを生かして体重が軽いことによる不利をいくらか補えるでしょう。Δt はたとえば 8 秒とします。

****

List 1:fgrep.c
List 2:chdir_by_filer.c
List 3:fgrep_.bat

```
/* fgrep program for MBCS(code page:932)                                    */
/* fgrep.c                                                                  */
/* by Morio Kikuchi 2018.1.1                                                */
/* WINDOW SYSTEM:Windows                                                    */
/* COMPILER:Visual C++ 4.0                                                  */
/* COMMANDLINE:cl /Fefgrep fgrep.c /link user32.lib gdi32.lib imm32.lib */
/* COMPILER:Open Watcom C/C++ 1.4                                           */
/* COMMANDLINE:wcc386 -w -j fgrep.c                                         */
/*             wlink file fgrep name fgrep library imm32.lib                */
/* 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 <windows.h>
#include <stdio.h>
#include <time.h>
#include <fcntl.h>
#include <sys/stat.h>

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

#define VGACOLORS 16
#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 ASIZE (MAX_PATH+1)
#define ASIZEM MAX_PATH
#define GKS GetKeyState
```

```
#define GKS_ GetKeyState
#define INVERT SRCINVERT
#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 XDSDY dv
#define FCSRSIZE 36                 /* cursorsize in filer */
#define SPCNUM 2                    /* spacecheck in filer */
#define SPCAFTER 8                  /* spaces after <DIR> */
#define SPC1 (0x81)                 /* 1st byte of full space */
#define SPC2 (0x40)                 /* 2nd byte of full space */
#define DSHIFT_2 2
#define NOTEKBD /*0*/1              /* Shift+BS=Del */
#define REP_Q_pl 0

int XRESO,YRESO,WB,COLUMN,ROW_L,ROW_S,ROW,FMAX,DJ,UDX,UDY,dh,dv,
    CSRDY,FAMILY,fontname,ACTIVE,INACTIVE,RTC,RETURN,TABCOLOR,CC,CSRCOLOR,
    CSRCOLOR_FILER,TAB_c=1,AINDENT,DX_FRAME,DY_FRAME,DY_CAPTION,DY_MENU,
    DY_TOOLBAR,RIGHT_M,LEFT_m,AVMEMDENO;
unsigned char *FONT;

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,returnflag,dirflag,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,
     overwriteflag,insorover,Flag_k,beginjumpflag,linelength_new,flag_REP_Q_pl,
     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,
    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,icsr_cfg,jcsr_cfg,icsr_string,jcsr_string,csrcolor,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,mfsize,kmax_ml,k_g;
```

```
double both=2;

char *openmode,*editflag,cc[]="@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\\]^_",fname_bg[ASIZE];
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*/10],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 immflag,compflag,dbflag,imeendflag,imm_restart_flag;
long dbsize,dbcount;

typedef struct {
/*int*/unsigned char red,green,blue;} srgb;  /* each of them <= 255 */
srgb irgb[VGACOLORS];
typedef struct {
unsigned long back_;int back,fore;} bf;
bf bfset[]={{WHITENESS,15,0},{BLACKNESS,0,15}};
typedef struct {
char flag;int number;} fs;
fs *fstack;
typedef struct {
int fn,icsr,jcsr;long firstk;} ft;
ft *ftable,ft_tmp;
typedef struct {
unsigned char dir[ASIZE],file[ASIZE];int rtn;} df;

HINSTANCE hinstance;
HWND hwnd;
HDC hdcdisplay,hdctmp1,hdctmp3;
HBITMAP hbitmap1,hbitmap3;
PAINTSTRUCT ps;
HFONT hfont;
HBRUSH hbrush;
HIMC himc,himc_;
COMPOSITIONFORM myime;
LOGFONT myimefont;
POINT point;

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),swap_BL(char),
    close_file(void),close_all(void),write_3vals(int),read_3vals(int),
    insertion_dk(char,long),nest_free(void),save_all(char),clear_topp(void),
    deletion_dk(void),deletion_dk_lump(void),BL(void),YKP(char),YKP_word(char),
    to_stack(unsigned char),to_stack_2b(unsigned char,unsigned char),frees(void),
    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),
    extraline(char),find_0x1a(char),refind(char),breaks(char),
    message(int,char),puts_message(int),puts_(int,int,unsigned char *),
    putstr(char,int,int,unsigned 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),indicator(char),
    csr_column_home(void),csr_column_end(void),csr_row_home(void),
    csr_down(void),csr_up(void),csr_right(void),tailcheck(void),find_word(char),
    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),copy_string(void),
    csr_to_1(void),csr_to_1_BL(char),edit_cfg(void),move_and_paste(void),
    string_visible(void),file_attri(void),overwrite(void),FILE_filename(void),
    memcpy_(unsigned char*,long,unsigned char*,long,long),
    half_word(char),restore_in_PAINT(void),copy_cfg(void),prompt_cq(char),
    scan_BL(void),FILE_jump(char),hcentering(char),restore_another(void),
    csr_row_end(void),within_linemax(void),dlgproc_REP_Quick(char,char*),

    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),
    csr_right_dialog(void),left_keydowns_dialog(void),

    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),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),memory(char),
    text_to_file(char,char),file_to_text(void),scroll_down(char),scroll_up(char),
    csr_left(void),return_is(int),initgraph_(void),insertion(unsigned char),
    wordcheck(char,long),wordcheck_kana(char,long),arraycheck(void),
    wordcheck_unvisible(char,long),wordcheck_2bytes(char,char,long),
    spacecheck(long,long),notspacecheck(char,LPSTR),while_puts_thepart(long,long),
    linestringcheck(void),read_cfg(int),getTAB(int),fgrep(unsigned char *),
    make_list(unsigned char *,unsigned char *),ishead_buf(unsigned char *,long),
    ishead(long),ishead_(long,long),ishead_ac(long),

    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),
     while_puts_dline(long,long),while_puts_firstk(long,long),
     while_puts_linenumber(long,long),

     gethead_dialog(char,long);

void imm_pause(void),imm_restart(void),imm_check(void),imm_close(void),
     WM_func_CHAR(WPARAM),WM_funcIME_CHAR(WPARAM),
     WM_funcIME_STARTCOMPOSITION(void),WM_funcIME_COMPOSITION(LPARAM),
     WM_funcIME_ENDCOMPOSITION(void);
COLORREF PALETTE(int color);

LRESULT CALLBACK wndproc_by_kbhit_(HWND,UINT,WPARAM,LPARAM);
int wndproc_filer(HWND,UINT,WPARAM,LPARAM),wndproc_ref(HWND,UINT,WPARAM,LPARAM),
    wndproc_deleted(HWND,UINT,WPARAM,LPARAM),wndproc_BL(HWND,UINT,WPARAM,LPARAM),
    wndproc(HWND,UINT,WPARAM,LPARAM);

void mnuproc_MULTIFILE(char *);
void mnuproc_REP(char *);
void dlgproc_OPEN(char);
void dlgproc_DRIVE(void);
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;

getcwd(fname_bg,ASIZE);

initgraph_();
refill=1;

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]);
}

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{
if(len>1 && isleadbyte(buf[len-2])==1 && ishead_buf(buf,len-2)==0)
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 APIENTRY WinMain(HINSTANCE hinst,HINSTANCE hinst_prev,LPSTR args,int show)
{
int argc/*,length*/;
unsigned char oldstring[ASIZE];
```

```
hinstance=hinst/*GetModuleHandle(NULL)*/;
if(FF_2/2) getcwd(fname_bg,ASIZE);

initgraph_();
refill=1;

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

if(strlen(args)==0){
/* 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{
/*fprintf_(0,0,args,fname);*/
argc=notspacecheck(0,args);

/*length=notspacecheck(1,args);*/
/*memcpy_(&array[0],0,&args[0],start_of_arg,length);*/
strcpy(array,nsc[0]);
/*array[length]='\0';*/

if((arraycheckflag=arraycheck())>1) {use_filer();argc=1;}

if(arraycheck()>0) {puts_mline(0,"Reinput a filename.");goto start;}
```

```
else strcpy(fname,array);

if(argc>1 && /*(length=*/notspacecheck(2,args)/*)*/<11){
/*memcpy_(&linestring[0],0,&args[0],start_of_arg,length);*/
strcpy(linestring,nsc[1]);
/*linestring[length]='\0';*/
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();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 */
}/**else(strlen(args))**/

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

mainroop();

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


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

len=strlen(plc);

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

```
if(plc[i]=='/') plc[i]='\\';
i++;

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]!='\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);
/*printf(" %s\n",array);*/
arraycheck();
/*printf(" %s\n",array);
getch();*/
strcpy(dfset.dir,array);

dfset.rtn=0;

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


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

sz=strlen(str);
if(sz>4){
strcpy(buf,&str[sz-4]);
if(stricmp(".exe",buf)==0) return 2;
if(stricmp(".obj",buf)==0) return 2;
if(stricmp(".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 %c\n",fn,p[fn][0]);*/
reference((char)((wsearch>0)?0:1));

return 0;
}/** fgrep **/


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 fprintf_(int cpage,int len,char* str1,char* str2)
{
FILE *fp;

fp=fopen("\\cpage.bin","ab");

fprintf(fp,"%d  %d  %s  %s\n",cpage,len,str1,str2);

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


void REP_Quick_pl(void)
{
int i,begin,end;
char str[ASIZE],page[ASIZE];
```

```
monitorline(1);

flag_REP_Q_pl=1;

dlgproc_REP_Quick(0,
"------------------------------------------------------------------------");

begin=1;end=500;

for(i=begin;i<=end;i++){
itoa(i,page,10);
sprintf(str,"%s%s%c%c%c%c"," Page ",page,0x0d,0x0a,0x0d,0x0a);  /* 0x0d:^M, 0x0a:^J */
dlgproc_REP_Quick(0,str);
}

flag_REP_Q_pl=0;
beep(50);

page_firstk(firstk);
csr();BitBltflag=/*1*/2;  /* BitBltflag=2:else{} in wndproc() */
}/** REP_Quick_pl **/


void dlgproc_REP_Quick(char reffunc,char *string)
{
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;
```

```
/*else */if(/*dialogflag==2*/1){                  /* job */
dialogflag=0;
dialogflag_REF=0;
icsr=icsr_old;jcsr=jcsr_old;

strcpy(rep_s,string);
strcpy(rep_s_,"");

allflag=1;direction=1;
lumpflag=1;

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);*/
memcpy_(&ptmp_rep[0],0,&p[fn][0],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);*/
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);*/
memcpy_(&p[fn][0],0,&ptmp_rep[0],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.");
csr();BitBltflag=2;*/
free(ptmp_rep);
}/**if(ptmp_rep)**/
else{
imm_restart();
message(7,2);
allflag=0;lumpflag=0;
page_firstk(firstk);
}/**else(ptmp_rep)**/

end:{}
}
}/** dlgproc_REP_Quick **/


void memcpy_(unsigned char *p_dst,long k_dst,unsigned char *p_src,long k_src,long sz)
{
long jump,i;

jump=k_dst-k_src;

if(jump>0){  /* up */
for(i=sz-1;i>=0;i--) p_dst[k_dst+i]=p_src[k_src+i];
}
else if(jump<0){  /* down */
for(i=0;i<=sz-1;i++) p_dst[k_dst+i]=p_src[k_src+i];
}
else if(jump==0){  /* parallel */
for(i=0;i<=sz-1;i++) p_dst[k_dst+i]=p_src[k_src+i];
}
}/** memcpy_ **/


void beep(long millisecond)
{
if(flag_REP_Q_pl) return;
```

```
Beep(888,millisecond);
}/** beep **/


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 **/


#if REP_Q_pl==0
void Replace(void)
{
char function_old;

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

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

```
#endif


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


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

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

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

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

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

  if(GKS_(VK_SHIFT)<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,unsigned char *str)
{
while_puts_show_str(flag,0,x,y,str);
}/** putstr */


int notspacecheck(char flag,LPSTR 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 **/


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]!=SPC1 || p[fn][member+1]!=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]!=SPC1 || p[fn][member+1]!=SPC2) break;}
else{}

csr_right();
}

k_left=member;

length=while_puts_thepart(k_left,k_right);
/*memcpy(&buf_line[0],&p[fn][k_left],k_right-k_left+1);*/
memcpy_(&buf_line[0],0,&p[fn][0],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)
/*memcpy(&p[fn][k+dk],&p[fn][k],kmax[fn]-dk-k+1);*/
memcpy_(&p[fn][0],k+dk,&p[fn][0],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]!=SPC1 || p[fn][member+1]!=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]!=SPC1 || p[fn][member+1]!=SPC2) break;}
else{}

csr_right();
}

k_left=member;

if((length=while_puts_thepart(k_left,k_right))==COLUMN+1) return;
/*memcpy(&buf_line[0],&p[fn][k_left],k_right-k_left+1);*/
memcpy_(&buf_line[0],0,&p[fn][0],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)
/*memcpy(&p[fn][k+dk],&p[fn][k],kmax[fn]-dk-k+1);*/
memcpy_(&p[fn][0],k+dk,&p[fn][0],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;

/*memcpy(&p[fn][k_0+dk_centering],&buf_line[0],k_right-k_left+1);*/
memcpy_(&p[fn][0],k_0+dk_centering,&buf_line[0],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);
/*fprintf_(k1,k2,"test","");*/
dk_deletion=k2-k1;
if(dk_deletion>0){
/*memcpy(&p[fn][k1],&p[fn][k2],kmax[fn]-k2+1);*/
memcpy_(&p[fn][0],k1,&p[fn][0],k2,kmax[fn]-k2+1);
kmax[fn]-=dk_deletion;

kmax[fn]+=1;
/*memcpy(&p[fn][k1+1],&p[fn][k1],kmax[fn]-1-k1+1);*/
memcpy_(&p[fn][0],k1+1,&p[fn][0],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){
  charflag=0;charcode=0;}
else if(GKS('N')<0){
  charflag=0;charcode=1;}
else if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){
  charflag=0;charcode=2;}
else{
  charflag=0;charcode=3;}
}/** keydowns_f2 **/


void restore_in_PAINT(void)
{
ValidateRect(hwnd,NULL);

if(compflag){
restore_3(0);
}/**if(compflag)**/
else if(imeendflag){
imeendflag=0;
restore_3(1);
}/**else if(imeendflag)**/
else{
/*restoreflag=1;*/
restore_3(1);
/*restoreflag=0;*/
}/**else(compflag,imeendflag)**/
}/** restore_in_PAINT **/


void imm_check(void)
{
himc_=ImmGetContext(hwnd);
if(immflag==1 && ImmGetOpenStatus(himc_)==TRUE) immflag=/*0*/2;
ImmReleaseContext(hwnd,himc_);
}/** imm_check **/


void imm_close(void)
{
himc_=ImmGetContext(hwnd);
if(ImmGetOpenStatus(himc_)==TRUE) ImmSetOpenStatus(himc_,FALSE);
```

```
immflag=0;
ImmReleaseContext(hwnd,himc_);
}/** imm_close **/


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


void imm_pause(void)
{
himc_=ImmGetContext(hwnd);
if(ImmGetOpenStatus(himc_)==TRUE){
immflag=1;ImmSetOpenStatus(himc_,FALSE);imm_restart_flag=1;
}
else imm_restart_flag=0;
ImmReleaseContext(hwnd,himc_);
}/** imm_pause **/


void imm_restart(void)
{
himc_=ImmGetContext(hwnd);
if(immflag==1 && ImmGetOpenStatus(himc_)==FALSE){
immflag=/*0*/2;ImmSetOpenStatus(himc_,TRUE);
}
else immflag=0;
/*imm_restart_flag=0;*/                  /* no problem ? */
ImmReleaseContext(hwnd,himc_);
}/** imm_restart **/


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_buf(unsigned char *buf,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(0,buf[k],member,-1);

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


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];
/*memcpy(&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];
/*memcpy(&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;

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];
/*memcpy(&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();

end_:{}
}/** 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);
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];
/*memcpy(&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];
/*memcpy(&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,FALSE)==TRUE)
puts_mline(0,"Default setup");
}/** copy_cfg **/


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 && editflag[fn]==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 **/


int make_list(unsigned char *home,unsigned char *fstr)
{
int flag;
long kmax_list=0,kceil_list=ASIZE*2,dk_list;
unsigned char *plist,*alloctmp,buf[ASIZE],buf_[ASIZE];
HANDLE hfirst;
WIN32_FIND_DATA fd;
BOOL bool;
SYSTEMTIME sysTime;
FILETIME fLocTime;

#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{
if(flag>1 && isleadbyte(buf_[flag-2])==1 && ishead_buf(buf_,flag-2)==0)
strcat(buf_,"\\*.*");
else strcat(buf_,"*.*");
}
}/**if(strlen(fstr))**/
else{
strcat(buf_,fstr);
/*printf(" %s\n",buf_);*/
}/**else(strlen(fstr))**/
#else
```

```
getcwd(buf_,ASIZE);
if(FF_2/2) strcpy(fname_bg,buf_);
if(buf_[3]!='\0') strcat(buf_,"\\*.*");else strcat(buf_,"*.*");
#endif

hfirst=FindFirstFile(buf_,&fd);
if(hfirst==INVALID_HANDLE_VALUE) return 2;
bool=TRUE;

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

count_dir=count_file=0;

while(bool==TRUE){
if((fd.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY)!=0){
FileTimeToLocalFileTime(&fd.ftLastWriteTime,&fLocTime);
FileTimeToSystemTime(&fLocTime,&sysTime);

dk_list=10+2+5+7+5+8+strlen(fd.cFileName)+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;
}

sprintf(&plist[kmax_list],"%04d/%02d/%02d  %02d:%02d       <DIR>         %s\n",
        sysTime.wYear,sysTime.wMonth,sysTime.wDay,sysTime.wHour,sysTime.wMinute,
        fd.cFileName);

kmax_list+=dk_list;
count_dir++;

#if GRP_or_EDT==0
if(to_sub==1){
if(strcmp(fd.cFileName,".")==0) goto next;
if(strcmp(fd.cFileName,"..")==0) goto next;
strcpy(array,fd.cFileName);
strcat(array,"\\");
arraycheck();
chdir(array);
make_list(array,fstr);
chdir(buf);
}
#endif
```

```
}/**if(fd)**/

next:
bool=FindNextFile(hfirst,&fd);
}

FindClose(hfirst);

hfirst=FindFirstFile(buf_,&fd);
bool=TRUE;

while(bool==TRUE){
if((fd.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY)==0){
FileTimeToLocalFileTime(&fd.ftLastWriteTime,&fLocTime);
FileTimeToSystemTime(&fLocTime,&sysTime);

dk_list=10+2+5+2+16+2+strlen(fd.cFileName)+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;
}

sprintf(&plist[kmax_list],"%04d/%02d/%02d  %02d:%02d  %16ld  %s\n",
        sysTime.wYear,sysTime.wMonth,sysTime.wDay,sysTime.wHour,sysTime.wMinute,
        fd.nFileSizeLow,fd.cFileName);

kmax_list+=dk_list;
count_file++;

#if GRP_or_EDT==0
if((fd.dwFileAttributes & FILE_ATTRIBUTE_ARCHIVE)!=0 &&
   (fd.dwFileAttributes & FILE_ATTRIBUTE_SYSTEM)==0 &&
   (fd.dwFileAttributes & FILE_ATTRIBUTE_HIDDEN)==0){
flag=fgrep(fd.cFileName);
if(flag==0){
free(p[fn]);/*free(ptmp);*/
fload_failed();
}
else if(flag==1){
fload_failed();
}
}
#endif
}/**if(fd)**/
```

```
bool=FindNextFile(hfirst,&fd);
}

FindClose(hfirst);

#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_:
FindClose(hfirst);

return 1;
}/** make_list **/


void filer(void)
{
char function_old;
char flag_,flag_open,flag_list;
int dm,length;
long k,dk_auto;
long member,member_,member_RETURN,member_Colon;
unsigned char buf[ASIZE],buf_[ASIZE],home[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,"dir /a /ogn > ");
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_);

/*if(buf_[3]!='\0') YKP(0);*/
if(buf_[3]!='\0') count_dir--;

dk_auto=length;
strcat(buf_,"\n");dk_auto++;
sprintf(count,"Dir(s):%ld  File(s):%ld\n",count_dir+1,count_file);
strcat(buf_,count);dk_auto+=strlen(count);

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

```
if(buf_[3]!='\n') jcsr++;
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){                           /* RETURN */
if(member==kmax[fn]) goto end;
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) goto end;
if(p[fn][member_]=='\n' && ishead(member_)==0) {member_++;break;}

member_--;
}

member_Start=member_;*/

member_=member;
dirflag=0;
while(1){                          /* Colon */
if(member_==0) goto end;
if(p[fn][member_]==':'/* && ishead(member_)==0*/) {/*member_++;*/break;}
if(p[fn][member_]=='<') dirflag=1;

member_--;
}

member_Colon=member_;

/*if(!spacenum){
if(spacecheck(member_Start,member_Colon)==1) spacenum=2;
else spacenum=1;
}*/

while(1){
if(member==0 || member<member_Colon) goto end;
if(p[fn][member]==' ' && spacecheck(member_Colon,member)==/*spacenum*/SPCNUM){
if(dirflag==0) {if(spaces==2) break;}
else {if(spaces==SPCAFTER) break;}
}

member--;
}

dm=member_RETURN-member-1;
/*dm=member_RETURN-member-1-1;*/      /* for 0x0d(
 by dir,sort) */

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]!='\0') strcat(buf_,"\\");
strcat(buf_,buf);

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((cdflag=chdir(buf))!=0 && dirflag!=1 && access(buf,4)==0){
/*getcwd(buf_,ASIZE);*/buf_[length]='\0';  /* restore */
if(buf_[3]!='\0') strcat(buf_,"\\");
strcat(buf_,buf);

strcpy(array,buf_);

if(refill==-2) {firstk_filer=firstk;icsr_filer=icsr;jcsr_filer=jcsr;/*cdflag=-1;*/}
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;}
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 **/


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

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

while(1){
if(checkflag==1 && p[fn][member_]==' '){
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 **/


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

tailcheck();

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

```
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;

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)**/
}/** YKP_word **/


void half_word(char flag)
```

```
{
lumpflag=1;

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

lumpflag_global=1/*0*/;
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/*0*/;
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==2){                         /* kana */
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==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_cut=dk;
paste=2;okflag_BL=1;
cut=0;
deletion_dk();}}
}/** half_line **/
/* YKP() */
```

```
/*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();}}*/


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(
/*type!=3 || */p[fn][member]!=SPC1 || p[fn][member+1]!=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(
/*type!=3 || */p[fn][member]!=SPC1 || p[fn][member+1]!=SPC2  /* !full space */
)
{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(
type!=3 || (p[fn][member]<0x88 && (p[fn][member]!=0x81 || p[fn][member+1]!=0x58))
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
}
else if(flag==1){
if(
type!=3 || ((p[fn][member]!=0x83 || p[fn][member+1]>0x96)  /* !katakana */
            && (p[fn][member]!=0x81 || p[fn][member+1]!=0x5B))
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
}
else if(flag==2){
if(
type!=3 || p[fn][member]!=0x82 || p[fn][member+1]<0x9F  /* !hiragana */
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
}
else if(flag==3){
```

```
if(
type!=3 || p[fn][member]!=0x82 || p[fn][member+1]>0x9A  /* !alphabet, !figure */
        || member==kmax[fn]
)
{k_to=member;break;}                /* k_right */
}
else{
if(
type!=3 || p[fn][member]!=0x83 || p[fn][member+1]<0x9F  /* !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]<0x88 && (p[fn][member]!=0x81 || p[fn][member+1]!=0x58))
)
{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]!=0x83 || p[fn][member+1]>0x96)  /* !katakana */
            && (p[fn][member]!=0x81 || p[fn][member+1]!=0x5B))
)
{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]!=0x82 || p[fn][member+1]<0x9F  /* !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]!=0x82 || p[fn][member+1]>0x9A  /* !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]!=0x83 || p[fn][member+1]<0x9F  /* !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)
{
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++;                           /* 1byte */

type=gettype_p(member);
if(
type!=0 || p[fn][member]<0xA6 || p[fn][member]>0xDF  /* !kana */
        || 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]<0xA6 || p[fn][member]>0xDF  /* !kana */
)
{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;
}/** 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), kana, 2bytes */
if(
type==0 && p[fn][member]>=0xA6 && p[fn][member]<=0xDF  /* kana */
)
wordcheck_kana(operation,k);
else if(
type==3 && (p[fn][member]>=0x88 || (p[fn][member]==0x81 && p[fn][member+1]==0x58))
)
wordcheck_2bytes(0,operation,k);
else if(
type==3 && ((p[fn][member]==0x83 && p[fn][member+1]<=0x96)  /* katakana */
            || (p[fn][member]==0x81 && p[fn][member+1]==0x5B))
)
wordcheck_2bytes(1,operation,k);
else if(
type==3 && p[fn][member]==0x82 && p[fn][member+1]>=0x9F  /* hiragana */
)
wordcheck_2bytes(2,operation,k);
else if(
```

```
type==3 && p[fn][member]==0x82 && p[fn][member+1]<=0x9A  /* alphabet, figure */
)
wordcheck_2bytes(3,operation,k);
else if(
type==3 && p[fn][member]==0x83 && p[fn][member+1]>=0x9F  /* 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]==/*0x81*/SPC1 && p[fn][member+1]==/*0x40*/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)
{
MSG msg;

/*while(1){
kbhit_();*/
while(GetMessage(&msg,NULL,0,0)){
TranslateMessage(&msg);
DispatchMessage(&msg);
if(refill!=1) break;

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

  if(/*nest<NEST*/1){
  nest++;
  function=1;if(GKS_(VK_CONTROL)<0 && fn!=FMAX-1) u_s_flag=1;else u_s_flag=0;
  if(GKS_(VK_SHIFT)<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)**/
}/**while(GetMessage)**/
}/** 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,
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(single byte) */
else
bitblt(-3,0*UDX,0*UDY,UDX*2,CSRDY,
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(double byte,1st) */
}
else
bitblt(-3,0*UDX,0*UDY,UDX*2,CSRDY,
         (icsr-1+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(double byte,2nd) */
}/**if(icsr)**/
else{
bitblt(-3,0*UDX,0*UDY,UDX,CSRDY,
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(over return) */
}/**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,
         (icsr+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);  /* dialog(single byte) */
else
bitblt(-3,0*UDX,0*UDY,UDX*2,CSRDY,
         (icsr+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);
}
else
bitblt(-3,0*UDX,0*UDY,UDX*2,CSRDY,
```

```
         (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,
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(single byte) */
else
bitblt(3,0*UDX,0*UDY,UDX*2,CSRDY,
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(double byte,1st) */
```

```
}
else
bitblt(3,0*UDX,0*UDY,UDX*2,CSRDY,
         (icsr-1+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(double byte,2nd) */
}/**if(icsr)**/
else{
bitblt(3,0*UDX,0*UDY,UDX,CSRDY,
         (icsr+DI)*UDX,(jcsr+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(over return) */
}/**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,
         (icsr+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);  /* dialog(single byte) */
else
bitblt(3,0*UDX,0*UDY,UDX*2,CSRDY,
         (icsr+DI_d)*UDX,(jcsr+DJ_d)*UDY+(UDY-CSRDY)+dy);
}
else
bitblt(3,0*UDX,0*UDY,UDX*2,CSRDY,
         (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 ceiling_theline(void)
{
long ddline;

within_linemax();

ddline=get_firstk(topp[jcsr],0);      /* ROW/2 <-> jcsr_ini */
if(ddline<0) jcsr=ddline+0;
```

```
else jcsr=0;
page_firstk(firstk);
}/** ceiling_theline **/


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 **/


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)**/
/*fprintf_(direction,0,"NEXT","");*/

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){
if(array[i]=='/') array[i]='\\';
i++;

if(i==length) break;
}

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

type=gettype_ac((long)i);
if(type<=2){
if(array[i]=='\\' && array[i+1]=='\\') {strcpy(&array[i],&array[i+1]);length--;}
else i++;
}
else{
i+=2;
}
}

if(array[0]!='\\' && array[1]!=':'){
```

```
getcwd(tmp,ASIZE);
strcat(tmp,"\\");
strcat(tmp,array);
strcpy(array,tmp);
}
else if(length>=3
        && ((array[0]>='A' && array[0]<='Z')||(array[0]>='a' && array[0]<='z'))
        && array[1]==':' && array[2]!='\\'){
getcwd(buf_b,ASIZE);
length=lstrlen(array);
strncpy(&tmp[0],&array[0],2);tmp[2]='\0';
chdir(tmp);
getcwd(buf_b2,ASIZE);lstrcpy(tmp,buf_b2);
strcat(tmp,"\\");
length_=strlen(tmp);
strncpy(&tmp[length_],&array[2],length-2);tmp[length_+length-2]='\0';
chdir(buf_b);
lstrcpy(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 */

i=length-1;
while(1){
if(array[i]=='\\' && ishead_ac((long)i)==0) break;
i--;if(i<0) break;
}

strcpy(tmp,array);
/*puts_mline(0,array);puts_mline_flag=0;*/

if(i>-1){                           /* '\\' exists. */
pspace=&tmp[i];
pspace++;
if(*pspace!='\0'){
strcpy(tmp_,pspace);                    /* tmp_ : filename */
*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{                               /* path and filename */
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(i)**/
else{                               /* '\\' does not exist. */
/*beep(50);*/
getcwd(fullpath,ASIZE);
strcat(fullpath,"\\");
strcat(fullpath,tmp);
strcpy(array,fullpath);
val=0;goto end;                             /* OK_file or unaccessible_directory
                                               in home_global_GCD */
}/**else(i)**/

end:
chdir(curdir_m);
end_:

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

type=gettype_ac((long)i);
if(type<=2){
if(array[i]=='\\' && array[i+1]=='\\') {strcpy(&array[i],&array[i+1]);length--;}
else i++;
}
else{
i+=2;
}
}

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";
#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_del[0],0,&stack_del[0],lessen,delsp-lessen);
/*memcpy(&stack_bs[0],&stack_bs[lessen],delsp-lessen);*/
memcpy_(&stack_bs[0],0,&stack_bs[0],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_del[0],0,&stack_del[0],lessen,delsp-lessen);
/*memcpy(&stack_bs[0],&stack_bs[lessen],delsp-lessen);*/
memcpy_(&stack_bs[0],0,&stack_bs[0],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_del[0],0,&stack_del[0],1,ASIZEM);
/*memcpy(&stack_bs[0],&stack_bs[1],ASIZEM);*/
memcpy_(&stack_bs[0],0,&stack_bs[0],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_del[0],0,&stack_del[0],1,ASIZEM);
/*memcpy(&stack_bs[0],&stack_bs[1],ASIZEM);*/
memcpy_(&stack_bs[0],0,&stack_bs[0],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: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,FALSE);
/*unlink(file_SA);*/
}
else if(length==ASIZEM-1){
strcpy(bak,home_global);
strcat(bak,"zzz.$$$");

CopyFile(file_SA,bak,FALSE);
/*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,FALSE);
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){
/*memcpy(&p[fn][k+dk_ins],&p[fn][k],(kmax[fn]-dk_ins)-(k)+1);*/
memcpy_(&p[fn][0],k+dk_ins,&p[fn][0],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);
/*memcpy(&p[fn][k],&p[fn][k+dk_ins],(kmax[fn])-(k+dk_ins)+1);*/
memcpy_(&p[fn][0],k,&p[fn][0],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) /*memcpy(&ptmp[0],&p[fn][k_from],dk);*/
                   memcpy_(&ptmp[0],0,&p[fn][0],k_from,dk);
else {dk=dk_old;flag_=1;}
}
else if(flag==1){                   /* ptmp_line, dk_line */
/*memcpy(&ptmp_line[0],&p[fn][k_from],dk_line);*/
memcpy_(&ptmp_line[0],0,&p[fn][0],k_from,dk_line);
}
else{                               /* ptmp_word, dk_word */
/*memcpy(&ptmp_word[0],&p[fn][k_from],dk_word);*/
memcpy_(&ptmp_word[0],0,&p[fn][0],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){
/*memcpy(&p[fn][k+dk],&p[fn][k],kmax[fn]-dk-k+1);*/
memcpy_(&p[fn][0],k+dk,&p[fn][0],k,kmax[fn]-dk-k+1);
p[fn][k]=charcode;
/****memcpy(&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){
/*memcpy(&p[fn][k+dk],&p[fn][k],kmax[fn]-dk-k+1);*/
memcpy_(&p[fn][0],k+dk,&p[fn][0],k,kmax[fn]-dk-k+1);
/*memcpy(&p[fn][k],&pdk[0],dk);*/
memcpy_(&p[fn][0],k,&pdk[0],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){
/*memcpy(&p[fn][k+dk_lump],&p[fn][k],kmax[fn]-dk_lump-k+1);*/
memcpy_(&p[fn][0],k+dk_lump,&p[fn][0],k,kmax[fn]-dk_lump-k+1);
/*memcpy(&p[fn][k],&rep_t_[0],dk_lump);*/
memcpy_(&p[fn][0],k,&rep_t_[0],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;
/*memcpy(&p[fn][k],&p[fn][k+dk],kmax[fn]-(k+dk)+1);*/
memcpy_(&p[fn][0],k,&p[fn][0],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;
/*memcpy(&p[fn][k],&p[fn][k+dk],kmax[fn]-(k+dk)+1);*/
memcpy_(&p[fn][0],k,&p[fn][0],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;
/*memcpy(&p[fn][k],&p[fn][k+dk],kmax[fn]-(k+dk)+1);*/
memcpy_(&p[fn][0],k,&p[fn][0],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;
/*memcpy(&p[fn][k],&p[fn][k+dk],kmax[fn]-(k+dk)+1);*/
memcpy_(&p[fn][0],k,&p[fn][0],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)
{
/*memcpy(&p[fn][k_from_rep],&p[fn][k_to_rep],kmax[fn]-k_to_rep+1);*/
memcpy_(&p[fn][0],k_from_rep,&p[fn][0],k_to_rep,kmax[fn]-k_to_rep+1);
kmax[fn]-=k_to_rep-k_from_rep;
}/** deletion_dk_lump **/


void deletion_dk(void)
{
/*memcpy(&p[fn][k_from],&p[fn][k_to],kmax[fn]-k_to+1);*/
memcpy_(&p[fn][0],k_from,&p[fn][0],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_ **/


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,line,TAB;
long k;

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*/int 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){
if(p[fn][k]==0x81 && (p[fn][k+1]==0x41 || p[fn][k+1]==0x42)) return 1;
else if(p[fn][k]==0x81 && (p[fn][k+1]==0x76 || p[fn][k+1]==0x78)) 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), kana, 2bytes */
```

```
if(
type==0 && p[fn][member]>=0xA6 && p[fn][member]<=0xDF  /* kana */
)
/*wordcheck_kana(operation,k);*/return 2;
else if(
type==3 && (p[fn][member]>=0x88 || (p[fn][member]==0x81 && p[fn][member+1]==0x58))
)
/*wordcheck_2bytes(0,operation,k);*/return 3;
else if(
type==3 && ((p[fn][member]==0x83 && p[fn][member+1]<=0x96)  /* katakana */
            || (p[fn][member]==0x81 && p[fn][member+1]==0x5B))
)
/*wordcheck_2bytes(1,operation,k);*/return 4;
else if(
type==3 && p[fn][member]==0x82 && p[fn][member+1]>=0x9F  /* hiragana */
)
/*wordcheck_2bytes(2,operation,k);*/return 5;
else if(
type==3 && p[fn][member]==0x82 && p[fn][member+1]<=0x9A  /* alphabet, figure */
)
/*wordcheck_2bytes(3,operation,k);*/return 6;
else if(
type==3 && (p[fn][member]==0x83 && p[fn][member+1]>=0x9F)  /* 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]==/*0x81*/SPC1 && p[fn][member+1]==/*0x40*/SPC2)/*)*/
)
/*wordcheck_unvisible(operation,k);*/return 8;
else if(
type==3 && p[fn][member]==0x81      /* symbol(2bytes) */
)
return 9;
else if(
type==3 && (p[fn][member]>=0x84 && p[fn][member]<=0x87)  /* 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<=0x7f) type=-1;
else type=3;

#else*/

/* code page:932(SJIS)-> */
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 */
/* <-code page:932(SJIS) */
/*else if(s1>=0x81 && s1<=0xfc && (s1<=0x9f || s1>=0xe0) &&
        s2>=0x40 && s2<=0xfc && s2!=0x7f) 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]>=0x81 && s[0]<=0xfc && (s[0]<=0x9f || s[0]>=0xe0) &&
        s[1]>=0x40 && s[1]<=0xfc && s[1]!=0x7f) type=3;
else type=-1;*/

return type;
}/** gettype_p **/


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,color;
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 */
else 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 && i==COLUMN){  /* Tab */
onceflag=0;tabflag=0;itab=0;k++;
ijlineflag=1;
```

```
}
else if(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 */
else 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) 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 && i==COLUMN){  /* Tab */
onceflag=0;tabflag=0;itab=0;k++;
ijlineflag=1;
}
else if(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) break;*/              /* ? */

k+=2/*DK*/;



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 */
else 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 */
else 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 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(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)
{
if(flag_REP_Q_pl) return;

firstk=max(k,0);if(firstk>kmax[fn]) get_firstk(kmax[fn],0);  /* protection */

if(lumpflag>0) {while_puts_show_(0,firstk);return;}
/*if(dbflag==1) {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,FALSE);
/*unlink(fname);*/
}
else if(length==ASIZEM-1){
strcpy(bak,home_global);
strcat(bak,"zzz.$$$");

CopyFile(fname,bak,FALSE);
/*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,FALSE);
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);           /* +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);
strncpy(data,&pcfg[i+length+2],k);
data[k]='\0';
```

```
return atoi(data);
}/** read_cfg **/


long get_av_memory(char flag,long mfsize)
{
static long val_1,val_2;
char str[ASIZE];
MEMORYSTATUS ms;
FILE *fp;

if(flag==0){
ms.dwLength=sizeof(MEMORYSTATUS);
GlobalMemoryStatus(&ms);

/*wsprintf(szmsg,"Total Phys. Mem: %ld\n"\
               "Avail Phys. Mem: %ld\n"\
               "Total Page File: %ld\n"\
               "Avail Page File: %ld\n"\
               "Total Virtual: %ld\n"\
               "Avail Virtual: %ld",
         ms.dwTotalPhys,
         ms.dwAvailPhys,
         ms.dwTotalPageFile,
         ms.dwAvailPageFile,
         ms.dwTotalVirtual,
         ms.dwAvailVirtual);*/

val_1=ms.dwAvailPhys/(1024L*1024);     /* MB */
val_2=ms.dwAvailVirtual/(1024L*1024);  /* MB */
}
else{
strcpy(str,home_global);
strcat(str,"mfsize.bin");

fp=fopen(str,"wb");

fprintf(fp,"available memory  = %ld + %ld = %ld[MB]\n",val_1,val_2,val_1+val_2);
fprintf(fp,"maximum filesize = %ld[MB]\n",mfsize);

fclose(fp);
}

return (val_1/*+val_2*/);
}/** get_av_memory **/
```

```
void setup(void)
{
int XRESO_MAX,YRESO_MAX,maxfiles;
long av_mem,fsize;
unsigned char buf[ASIZE],home[ASIZE];

GetCurrentDirectory(ASIZE,home_global_GCD);

if(chdir("c:\\ble")==0) {strcpy(home_global,"c:\\ble\\");chdir(home_global_GCD);}
else if(chdir("d:\\ble")==0) {strcpy(home_global,"d:\\ble\\");chdir(home_global_GCD);}
else if(chdir("e:\\ble")==0) {strcpy(home_global,"e:\\ble\\");chdir(home_global_GCD);}
else{
GetWindowsDirectory(buf,ASIZE);
strncpy(home_global,buf,3);         /* c:\, d:\, e:\ */
home_global[3]='\0';
}

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");

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=GetSystemMetrics(SM_CXSCREEN);
YRESO_MAX=GetSystemMetrics(SM_CYSCREEN);

DX_FRAME=GetSystemMetrics(SM_CXSIZEFRAME);
DY_FRAME=GetSystemMetrics(SM_CYSIZEFRAME);
DY_CAPTION=GetSystemMetrics(SM_CYCAPTION);
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=9;UDY=20;

     /*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;}*/
COLUMN=92;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;
fontname=0;
dh=/*0*/-2;
dv=/*0*/1;
l_s_flag=0;
tabspaces=1;
MOVEcsr=1;                          /* pm1 */
LEFT_m=60;
Flag_k=0;
AVMEMDENO=8;
mfsize=av_mem/AVMEMDENO;
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));  /* no 0x1a */
if(pcfg==NULL) {fclose(fp);goto start;}

fseek(fp,0L,SEEK_SET);
fread(pcfg,1,fsize,fp);
cfgmax=fsize/1-1;
fclose(fp);

/* 0 -> 3 : 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);

/* 4 : size of tab */
if((TAB_c=read_cfg(5))==-1000) TAB_c=8;
else TAB_c=max(min(TAB_c,COLUMN-1),1);

/* 5 : size of cursor */
if((CSRDY=read_cfg(6))==-1000) CSRDY=UDY;
else {if(CSRDY==0) CSRDY=UDY;else CSRDY=max(min(CSRDY,UDY),1);}

/* 6 -> 13 : 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*/;
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((fontname=read_cfg(17))==-1000) fontname=0;
else fontname=min(fontname,1);
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;
else mfsize=min(mfsize,av_mem/AVMEMDENO);
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)
{
int UDX_,UDY_,dx=0,dy=-2;
WNDCLASS wndclass;

setup();

#if GRP_or_EDT==1
initpalette();

UDX_=UDX+dx;
UDY_=UDY+/*dy*/dh;
UDY_=max(min(UDY_,64),8);
```

```
wndclass.hInstance    =hinstance;
wndclass.lpszClassName="BLECLASS";
wndclass.lpszMenuName ="BLEMENU";
wndclass.lpfnWndProc  =(WNDPROC)wndproc_by_kbhit_;
wndclass.style        =CS_HREDRAW | CS_VREDRAW;
wndclass.hIcon        =LoadIcon(hinstance,NULL);
wndclass.hCursor      =LoadCursor(NULL,IDC_ARROW);
wndclass.cbClsExtra   =0;
wndclass.cbWndExtra   =0;
if(WB==0)
wndclass.hbrBackground=GetStockObject(WHITE_BRUSH);
else
wndclass.hbrBackground=GetStockObject(BLACK_BRUSH);

if(RegisterClass(&wndclass)==0) exit(1);

hwnd=CreateWindow("BLECLASS","BLE",
                  WS_OVERLAPPEDWINDOW,
                  LEFT_m,0,XRESO,YRESO,
                  NULL,NULL,hinstance,NULL);
if(hwnd==NULL) {UnregisterClass("BLECLASS",hinstance);exit(1);}

/*ShowWindow(hwnd,show_);
UpdateWindow(hwnd);*/
SetWindowPos(hwnd,HWND_TOP,0,0,0,0,SWP_NOMOVE | SWP_NOSIZE);
ShowWindow(hwnd,SW_SHOWDEFAULT);

hdcdisplay=GetDC(hwnd);

hbitmap1=CreateCompatibleBitmap(hdcdisplay,XRESO,YRESO);
hbitmap3=CreateCompatibleBitmap(hdcdisplay,XRESO,UDY*3);

hdctmp1=CreateCompatibleDC(hdcdisplay);  /* text, dialog, menu */
hdctmp3=CreateCompatibleDC(hdcdisplay);  /* cursor */

SelectObject(hdctmp1,hbitmap1);
SelectObject(hdctmp3,hbitmap3);

SetBkMode(hdcdisplay,TRANSPARENT);
SetBkMode(hdctmp1,TRANSPARENT);
SetBkMode(hdctmp3,TRANSPARENT);

SetBkColor(hdcdisplay,PALETTE(bfset[WB].back));
SetBkColor(hdctmp1,PALETTE(bfset[WB].back));
SetBkColor(hdctmp3,PALETTE(bfset[WB].back));
```

```
if(fontname==0) {FAMILY=FF_ROMAN;FONT="MSMINCHO";}
else           {FAMILY=FF_MODERN;FONT="MSGOTHIC";}
hfont=CreateFont(UDY_,UDX_,0,0,
                 FW_NORMAL,0,0,0,
                 DEFAULT_CHARSET,OUT_DEFAULT_PRECIS,
                 CLIP_DEFAULT_PRECIS,DEFAULT_QUALITY,
                 FIXED_PITCH | FAMILY,NULL/*FONT*/);
SelectObject(hdcdisplay,hfont);
SelectObject(hdctmp1,hfont);

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)
{
frees();
free(ptmp);

#if GRP_or_EDT==1
DeleteObject(hfont);
DeleteObject(hbitmap1);
DeleteObject(hbitmap3);
DeleteDC(hdctmp1);
DeleteDC(hdctmp3);

ReleaseDC(hwnd,hdcdisplay);
DestroyWindow(hwnd);
```

```
UnregisterClass("BLECLASS",hinstance);

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;
}/** initpalette **/


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)
{
/*monitorline(0);*/
```

```
/*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 */

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)**/

/*putpixel(50,YRESO,0);*/
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;
/*setcsrcolor(csrcolor_tmp);*/
/*fprintf_(csrcolor_tmp,1,NULL,NULL);*/

if(flag_2nd==0){
if(gettype_p(k)!=3)
bitblt(-3,0*UDX,/*0*/2*UDY,UDX,CSRDY,
         (icsr_f+DI)*UDX,(jcsr_f+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(single byte) */
else
bitblt(-3,0*UDX,/*0*/2*UDY,UDX*2,CSRDY,
```

```
         (icsr_f+DI)*UDX,(jcsr_f+DJ)*UDY+(UDY-CSRDY)+dy);  /* text(double byte,1st) */
}
else
bitblt(-3,0*UDX,/*0*/2*UDY,UDX*2,CSRDY,
         (icsr_f-1+DI)*UDX,(jcsr_f+DJ)*UDY+(UDY-CSRDY)+dy);

/*setcsrcolor(csrcolor);*/              /* restore */
/*fprintf_(csrcolor,2,NULL,NULL);*/
}
}
}/**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"};

/*if(flag_REP_Q_pl) return;*/

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) 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;

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 (-> imm_pause()) */
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

SetTextColor(hdcdisplay,PALETTE(color));
SetTextColor(hdctmp1,PALETTE(color));
}/** setstccolor **/


void stc(char flag,int x,int y,unsigned char *str,int size)
{
#if GRP_or_EDT==0
return;
#endif

if(flag==1)
TextOut(hdctmp1,x,y+XDSDY,str,size);
else
TextOut(hdcdisplay,x,y+XDSDY,str,size);
}/** stc **/


void cleardevice_(char flag,int x,int y,int xsize,int ysize)
```

```
{
int dx;

if(cut==0) dx=0;
else dx=UDX;

if(flag==0)
PatBlt(hdcdisplay,x,y,xsize,ysize,bfset[WB].back_);
else if(flag==-1){
if(divisionnumber==0)
PatBlt(hdctmp1,dx,0,XRESO-dx,YRESO,bfset[WB].back_);
else if(divisionnumber==1)
PatBlt(hdctmp1,dx,0,XRESO-dx,(ROW+2)*UDY,bfset[WB].back_);
else
PatBlt(hdctmp1,dx,DJ*UDY,XRESO-dx,YRESO-DJ*UDY,bfset[WB].back_);  /* DJ = ROW+2 */
}
else if(flag==1)
PatBlt(hdctmp1,x,y,xsize,ysize,bfset[WB].back_);
else
PatBlt(hdctmp3,x,y,xsize,ysize,bfset[WB].back_);
}/** cleardevice_ **/


void paint(char flag,int x,int y,int xsize,int ysize,int color)
{
hbrush=CreateSolidBrush(PALETTE(color));

if(flag==0){
SelectObject(hdcdisplay,hbrush);
PatBlt(hdcdisplay,x,y,xsize,ysize,PATCOPY);
}
else if(flag==1){
SelectObject(hdctmp1,hbrush);
PatBlt(hdctmp1,x,y,xsize,ysize,PATCOPY);
}
else{
SelectObject(hdctmp3,hbrush);
PatBlt(hdctmp3,x,y,xsize,ysize,PATCOPY);
}

DeleteObject(hbrush);
}/** paint **/


void setcsrcolor(int color)
{
```

```
hbrush=CreateSolidBrush(PALETTE(color));
SelectObject(hdctmp3,hbrush);
PatBlt(hdctmp3,0,0,XRESO,/*YRESO*/UDY,PATCOPY);
DeleteObject(hbrush);
}/** setcsrcolor **/


COLORREF PALETTE(int color)
{
return RGB(irgb[color].red,irgb[color].green,irgb[color].blue);
}/** PALETTE **/


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)
PatBlt(hdctmp1,0,0,UDX,YRESO,bfset[WB].back_);
else if(divisionnumber==1)
PatBlt(hdctmp1,0,0,UDX,(ROW+2)*UDY,bfset[WB].back_);
else
PatBlt(hdctmp1,0,DJ*UDY,UDX,YRESO-DJ*UDY,bfset[WB].back_);  /* 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);
hbrush=CreateSolidBrush(PALETTE(ACTIVE));
SelectObject(hdctmp1,hbrush);
PatBlt(hdctmp1,0,(jcsr_1+DJ)*UDY,UDX,(jcsr_2-jcsr_1)*UDY,PATCOPY);
DeleteObject(hbrush);

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);
hbrush=CreateSolidBrush(PALETTE(ACTIVE));
SelectObject(hdctmp1,hbrush);
if(jcsr_f<ROW) djcsr=jcsr_2-jcsr_1+1;else djcsr=jcsr_2-jcsr_1;
PatBlt(hdctmp1,0,(jcsr_1+DJ)*UDY,UDX,djcsr*UDY,PATCOPY);
DeleteObject(hbrush);

indicationflag=1;
}
}/**else(cut)**/

skip:
```

```
BitBlt_indicator();
}/** indicator **/


void BitBlt_indicator(void)
{
int dy=DSHIFT_2;

if(divisionnumber==0)
BitBlt(hdcdisplay,0,0+dy,UDX,YRESO,
       hdctmp1,0,0,SRCCOPY);
else if(divisionnumber==1)
BitBlt(hdcdisplay,0,0+dy,UDX,(ROW+2)*UDY,
       hdctmp1,0,0,SRCCOPY);
else
BitBlt(hdcdisplay,0,DJ*UDY+dy,UDX,YRESO-DJ*UDY,  /* DJ = ROW+2 */
       hdctmp1,0,DJ*UDY,SRCCOPY);
}/** 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)
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp1,x,y,SRCCOPY);
else if(flag==3)                    /* flag = 3 */ /* for csr() */
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp1,/*x*/x_,/*y*/y_-dy_,SRCCOPY);
}/**if(bitbltflag)**/
else{
/*if(flag==0) {}
else */if(flag==1)
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp1,x,y,INVERT);
else if(flag==-3)              /* flag = -3 */ /* for csr_to_1(), for csr_to_1_BL() */
BitBlt(hdctmp1,x_,y_-dy_,xsize,ysize,
       hdctmp3,x,y,INVERT);         /* x, y = 0, 0 */
}/**else(bitbltflag)**/
}/** 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;  /* icsr:invalid */
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;  /* icsr:invalid */
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;

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)
{
STARTUPINFO sui;
PROCESS_INFORMATION pi;

if(systemflag) system(exefile);
else{
ZeroMemory(&sui,sizeof(STARTUPINFO));
ZeroMemory(&pi,sizeof(PROCESS_INFORMATION));
sui.cb=sizeof(STARTUPINFO);
sui.dwFlags=STARTF_USESHOWWINDOW;
sui.wShowWindow=SW_SHOWNORMAL;

CreateProcess(NULL,exefile,NULL,NULL,0,0,NULL,NULL,&sui,&pi);
}
}/** 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);*/

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();

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 **/


char *getdrive(void)
{
char i;
int drivesmax=26-2,drives=0;
char str[10],old[MAX_PATH+1];
static char p[27]="";

getcwd(old,MAX_PATH);

for(i=0;i<drivesmax;i++){
sprintf(str,"%c",i+'C');
strcat(str,":\\");
if(chdir(str)==0){
str[1]='\0';
strcat(p,str);

drives++;
}
}

chdir(old);

p[drives]='\0';

return p;
}/** *getdrive **/


void dlgproc_DRIVE(void)
{
int i,length;
int icsr_old,jcsr_old;
char title_d[32],drive[26+1+1];     /* +1+1:+0x1a+'\0' */
unsigned char GCD[MAX_PATH+1];
```

```
if(cut==2) csr_to_1_BL(1);csr_to_1();  /* write csr to hdctmp1 */

/*usflag=1;*/

icsr_old=icsr;jcsr_old=jcsr;

/*i=2;*/                                  /* from C */
/*while(1){
if(((GetLogicalDrives() >> i) & 1)==0) {i--;break;}
i++;
}*/

/*strcpy(title_d,"Drive <= ");*/
strcpy(title_d,"Drive");
/*length=strlen(title_d);
title_d[length]=(char)i+0x41;
title_d[length+1]='\0';*/

start:

dialogflag=1;

strcpy(drive,getdrive());
/*strncpy(drive,"ABCDEFGHIJKLMNOPQRSTUVWXYZ",i+1);
drive[i+1]='\0';*/
/*GetCurrentDirectory*/getcwd(GCD,MAX_PATH);
/*fprintf_(0,GCD,"");*/

length=strlen(drive);
for(i=0;i<length;i++){
if(strnicmp(&drive[i],&GCD[0],1)==0) break;  /* use i */
}
if(i==length){
dialogflag=0;
driveflag=0;
icsr=icsr_old;jcsr=jcsr_old;
page_firstk(firstk);
return;
}
/*fprintf_(icsr,"","");*/

strcpy(p_dialog,drive);        /* to p_dialog */
driveflag=1;
before_mainroop(title_d);

kmax_dialog=strlen(p_dialog)-1;
```

```
/*if(GCD[0]>=0x61 && GCD[0]<=0x7a) GCD[0]-=0x20;
icsr=GCD[0]-0x41;*/
icsr=i;

page_firstk_dialog(0);
csr();
mainroop();                          /* p_dialog */
after_mainroop();
strncpy(drive,&p_dialog[icsr],1);        /* to drive */
drive[1]='\0';

kmax_dialog=0;                       /* <-> clear_dialog() */
p_dialog[0]=0x1a;
p_dialog[1]='\0';

if(dialogflag==3){
dialogflag=0;
driveflag=0;
icsr=icsr_old;jcsr=jcsr_old;
/*BitBlt_full();*/page_firstk(firstk);
}

else if(dialogflag==2){                  /* job */
dialogflag=0;
icsr=icsr_old;jcsr=jcsr_old;

if(strlen(drive)==0) goto start;
if(drive[0]>=0x61 && drive[0]<=0x7a) drive[0]-=0x20;
if(drive[0]>=0x41 && drive[0]<=0x5a) {drive[1]='\0';strcat(drive,":");}
else goto start;

if(/*SetCurrentDirectory*/chdir(drive)==-1) goto start;

refill=0;charflag=0;charcode=2;
BitBltflag=2;
}
}/** dlgproc_DRIVE **/


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);
```

```
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_Colon;
unsigned char buf[ASIZE],buf_[ASIZE],home[ASIZE];
```

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

member_--;
}

member_Start=member_;*/

member_=member;
dirflag=0;
while(1){                          /* Colon */
if(member_==0) goto end;
if(p[fn][member_]==':'/* && ishead(member_)==0*/) {/*member_++;*/break;}
if(p[fn][member_]=='<') dirflag=1;

member_--;
}

member_Colon=member_;

/*if(!spacenum){
if(spacecheck(member_Start,member_Colon)==1) spacenum=2;
else spacenum=1;
}*/

while(1){
if(member==0 || member<member_Colon) goto end;
if(p[fn][member]==' ' && spacecheck(member_Colon,member)==/*spacenum*/SPCNUM){
if(dirflag==0) {if(spaces==2) break;}
else {if(spaces==SPCAFTER) break;}
}
```

```
member--;
}

dm=member_RETURN-member-1;
/*dm=member_RETURN-member-1-1;*/      /* for 0x0d */

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

getcwd(buf_,ASIZE);
if(buf_[3]!='\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(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_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;  /* charcode=0(Y),1(N),2(Esc, Pause),3(else) */
}

/*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);*/
memcpy_(&ptmp_rep[0],0,&p[fn][0],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);*/
memcpy_(&p[fn][0],0,&ptmp_rep[0],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_;

if(driveflag) return;

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;

if(driveflag) return 1;

tailcheck_dialog();
```

```
flag_=0;

kmax_dialog++;
if(kmax_dialog>ASIZEM-1) {beep(50);kmax_dialog--;flag_=1;}
else{
k=firstk_dialog+icsr;
/*memcpy(&p_dialog[k+1],&p_dialog[k],kmax_dialog-1-k+1);*/
memcpy_(&p_dialog[0],k+1,&p_dialog[0],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;

if(driveflag) return 1;

tailcheck_dialog();

k=firstk_dialog+icsr;
if(k==kmax_dialog) return 1;

type=gettype_dialog(k);

if(type<=2){
dk=1;
/*memcpy(&p_dialog[k],&p_dialog[k+dk],kmax_dialog-(k+dk)+1);*/
memcpy_(&p_dialog[0],k,&p_dialog[0],k+dk,kmax_dialog-(k+dk)+1);
kmax_dialog-=dk;
}/**if(type)**/
else if(type==3){
```

```
dk=2;
/*memcpy(&p_dialog[k],&p_dialog[k+dk],kmax_dialog-(k+dk)+1);*/
memcpy_(&p_dialog[0],k,&p_dialog[0],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)
{
if(driveflag) return;

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 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 ***************************/

void kbhit_(void)
{
MSG msg;

if(PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
/*if(GetMessage(&msg,NULL,0,0)){*/
TranslateMessage(&msg);
```

```
DispatchMessage(&msg);
}
}/** kbhit_ */


LRESULT CALLBACK wndproc_by_kbhit_(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
if(cut>0){
if(wndproc_BL(hwnd,umsg,wparam,lparam)!=0) return 1;
}
else if(filerflag==1){
if(wndproc_filer(hwnd,umsg,wparam,lparam)!=0) return 1;
}
else if(deletedflag==1){
if(wndproc_deleted(hwnd,umsg,wparam,lparam)!=0) return 1;
}
else if(refflag==1){
if(wndproc_ref(hwnd,umsg,wparam,lparam)!=0) return 1;
}
else{
if(wndproc(hwnd,umsg,wparam,lparam)!=0) return 1;
}

return DefWindowProc(hwnd,umsg,wparam,lparam);
}/** wndproc_by_kbhit_ **/


int wndproc_filer(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
char gotoflag;

if(umsg==WM_KEYDOWN){
BitBltflag=0;BitBltflag_=0;

/************************** menu keydowns -> ****************************/
/************************** <- menu keydowns ****************************/

/************************** dialog keydowns -> ****************************/

if(dialogflag>0){

imm_check();

if(immflag==2) immflag=0;
if(usflag==1) usflag=0;
```

```
if(compflag) return 1;

if(cqflag==2){
  BitBltflag_=2;
  goto end_dialog;}
if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_dialog;}

gotoflag=1;

if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){  /* added */
  dialogflag=3;refill=0;BitBltflag_=2;}
else if(GKS(VK_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(BitBltflag_==0) {BitBlt_dialog();csr();}
else if(BitBltflag_==1)             csr();
else{}

return 1;
}/**if(dialogflag)**/

/************************** <- dialog keydowns ****************************/

if(function==2){
imm_pause();
keydowns_f2();

return 1;
}

imm_pause();

if(immflag==2) immflag=0;
if(usflag==1) usflag=0;
```

```
if(compflag) return 1;

if(cqflag==6){
  /*BitBltflag=2;*/
  goto end_left;}

gotoflag=1;

/*9*/
if(GKS_(VK_SHIFT)<0 && GKS(VK_F4)<0){
  if(function==1) filerskip=1;
  refill=-2;/*if(GKS_(VK_SHIFT)<0) refill--;*/charflag=0;charcode=2;
  filer_execute=1;
  BitBltflag=2;}
else if(GKS(VK_F4)<0 || GKS('D')<0){
  dlgproc_DRIVE();}

else if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){
  if(function==0){
  refill=0;if(GKS_(VK_SHIFT)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}
  else{
  charflag=0;charcode=2;
  BitBltflag=2;}}
else if(GKS(VK_RETURN)<0){
  if(function==1) filerskip=1;
  refill=-2;if(GKS_(VK_SHIFT)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}

else if(GKS_(VK_CONTROL)<0 && GKS('Q')<0){
  cqflag=5;prompt_cq(2);}

else if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)<0 && GKS('F')<0){
  FILE_filename();}

else if(GKS(VK_F9)<0){
  Find(0);}

else {gotoflag=0;}

if(gotoflag==1) goto end;

end_left:
gotoflag=left_keydowns();

if(gotoflag==0) goto end_;  /* no job */
```

```
end:
if(topp[jcsr]<topp_floor) {/*beep(50)*/;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(umsg)**/
else if(umsg==WM_SYSKEYDOWN){
if(cqflag){
extraline(1);cqflag=0;
if(dialogflag>0 && imm_restart_flag==1) imm_restart();}
}/**else if(umsg)**/
else if(umsg==WM_CHAR){
  WM_func_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_CHAR){
  WM_funcIME_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_STARTCOMPOSITION){
  WM_funcIME_STARTCOMPOSITION();
}/**else if(umsg)**/
else if(umsg==WM_IME_COMPOSITION){
  WM_funcIME_COMPOSITION(lparam);
}/**else if(umsg)**/
else if(umsg==WM_IME_ENDCOMPOSITION){
  WM_funcIME_ENDCOMPOSITION();
}/**else if(umsg)**/
else if(umsg==WM_CLOSE){
imm_close();
breaks(0);

if(dialogflag>0){
dialogflag=3;refill=0;
}
else{
refill=0;if(GKS_(VK_SHIFT)<0) refill--;charflag=0;charcode=2;
}

return 1;
```

```
}/**else if(umsg)**/
else if(umsg==WM_PAINT){
restore_in_PAINT();

return 1;
}/**else if(umsg)**/
else{}

return 0;
}/** wndproc_filer **/


int wndproc_ref(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
char gotoflag;

if(umsg==WM_KEYDOWN){
BitBltflag=0;

/************************** menu keydowns -> ****************************/
/************************** <- menu keydowns ****************************/

/************************** dialog keydowns -> ****************************/
/************************** <- dialog keydowns ****************************/

if(function==2){
imm_pause();
keydowns_f2();

return 1;
}

if(immflag==2) immflag=0;
if(usflag==1) usflag=0;

if(compflag) return 1;

if(cqflag==2){
  BitBltflag=2;
  goto end;}
if(cqflag==6){
  /*BitBltflag=2;*/
  goto end_left;}

gotoflag=1;
```

```
/*9*/
if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){  /* added */
  refill=0;if(GKS_(VK_SHIFT)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}
/* 4if */
else if(GKS(VK_RETURN)<0){
  if(GKS_(VK_SHIFT)<0 || GKS_(VK_CONTROL)<0){
  if(Enter(1)==1) goto end;}
  else{
  refill=-2;charflag=0;charcode=2;
  BitBltflag=2;}}

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('I')<0){
  cqflag=1;prompt_cq(0);}
else if(GKS_(VK_CONTROL)<0 && GKS('Q')<0){
  cqflag=5;prompt_cq(2);}

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('Y')<0){
  YKP_word(0);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('K')<0){
  YKP_word(1);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('P')<0){
  YKP_word(2);}
else if(GKS_(VK_CONTROL)<0 && GKS('Y')<0){
  YKP(0);}
else if(GKS_(VK_CONTROL)<0 && GKS('K')<0){
  YKP(1);}
else if(GKS_(VK_CONTROL)<0 && GKS('P')<0){
  YKP(2);}

else if(GKS(VK_F6)<0){
  paste=0;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS_(VK_SHIFT)<0 && GKS(VK_F7)<0){
  paste=2;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS(VK_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) goto end_;  /* no job */
```

```
end:
if(BitBltflag==0)      {BitBlt_full();csr();}
else if(BitBltflag==1) {monitorline(1);csr();}
else{}

end_:
return 1;
}/**if(umsg)**/
else if(umsg==WM_SYSKEYDOWN){
if(cqflag) {extraline(1);cqflag=0;imm_restart();}
}/**else if(umsg)**/
else if(umsg==WM_CHAR){
  WM_func_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_CHAR){
  WM_funcIME_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_STARTCOMPOSITION){
  WM_funcIME_STARTCOMPOSITION();
}/**else if(umsg)**/
else if(umsg==WM_IME_COMPOSITION){
  WM_funcIME_COMPOSITION(lparam);
}/**else if(umsg)**/
else if(umsg==WM_IME_ENDCOMPOSITION){
  WM_funcIME_ENDCOMPOSITION();
}/**else if(umsg)**/
else if(umsg==WM_CLOSE){
imm_close();
breaks(0);

refill=0;charflag=0;charcode=2;

return 1;
}/**else if(umsg)**/
else if(umsg==WM_PAINT){
restore_in_PAINT();

return 1;
}/**else if(umsg)**/
else{}

return 0;
}/** wndproc_ref **/
```

```
int wndproc_deleted(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
char gotoflag;

if(umsg==WM_KEYDOWN){
BitBltflag=0;BitBltflag_=0;

/************************* menu keydowns -> ****************************/

if(menuflag>0){

imm_pause();

if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_menu;}

gotoflag=1;

if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){  /* added */
  menuflag=3;refill=0;BitBltflag_=2;}
else if(GKS(VK_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(BitBltflag_==0) {BitBlt_menu();csr();}
else if(BitBltflag_==1) csr();
else{}

return 1;
}/**if(menuflag)**/

/************************* <- menu keydowns ****************************/

/************************* dialog keydowns -> ****************************/

if(dialogflag>0){
```

```
if(immflag==2) immflag=0;
if(usflag==1) usflag=0;

if(compflag) return 1;

if(cqflag==2){
  BitBltflag_=2;
  goto end_dialog;}
if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_dialog;}

gotoflag=1;

if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){  /* added */
  dialogflag=3;refill=0;BitBltflag_=2;}
else if(GKS(VK_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(BitBltflag_==0) {BitBlt_dialog();csr();}
else if(BitBltflag_==1)             csr();
else{}

return 1;
}/**if(dialogflag)**/

/************************* <- dialog keydowns ****************************/

if(function==2){
imm_pause();
keydowns_f2();

return 1;
}

if(immflag==2) immflag=0;
if(usflag==1) usflag=0;
```

```
if(function==3) function=0;
if(function==4) function=1;

if(compflag) return 1;

if(cqflag==2){
  BitBltflag=2;
  goto end;}
if(cqflag==6){
  /*BitBltflag=2;*/
  goto end_left;}

gotoflag=1;

/*9*/
if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){  /* added */
  if(function==0){
  refill=0;if(GKS_(VK_SHIFT)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}
else{
  charflag=0;charcode=2;
  BitBltflag=2;}}
else if(GKS(VK_RETURN)<0){
  if(Enter(0)==1) goto end;}
/* 4if */
/*else if(GKS(VK_RETURN)<0){
  if(GKS_(VK_SHIFT)<0 || GKS_(VK_CONTROL)<0){
  if(Enter(1)==1) goto end;}
  else{
  refill=-2;charflag=0;charcode=2;
  BitBltflag=2;}}*/

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('I')<0){
  cqflag=1;prompt_cq(0);}
else if(GKS_(VK_CONTROL)<0 && GKS('Q')<0){
  cqflag=5;prompt_cq(2);}

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('Y')<0){
  YKP_word(0);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('K')<0){
  YKP_word(1);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('P')<0){
  YKP_word(2);}
else if(GKS_(VK_CONTROL)<0 && GKS('Y')<0){
  YKP(0);}
else if(GKS_(VK_CONTROL)<0 && GKS('K')<0){
```

```
  YKP(1);}
else if(GKS_(VK_CONTROL)<0 && GKS('P')<0){
  YKP(2);}

else if(GKS(VK_F6)<0){
  paste=0;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS_(VK_SHIFT)<0 && GKS(VK_F7)<0){
  paste=2;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS(VK_F7)<0){
  paste=1;cut=0;monitorline(1);BitBltflag=2;}

else if(GKS(VK_F5)<0){
  Replace();}
else if(GKS(VK_F9)<0){
  Find(0);}

else {gotoflag=0;}

if(gotoflag==1) goto end;

end_left:
gotoflag=left_keydowns();

if(gotoflag==0) goto end_;  /* no job */

end:
if(BitBltflag==0)      {BitBlt_full();csr();}
else if(BitBltflag==1) {monitorline(1);csr();}
else{}

end_:
return 1;
}/**if(umsg)**/
else if(umsg==WM_SYSKEYDOWN){
if(cqflag) {extraline(1);cqflag=0;imm_restart();}
}/**else if(umsg)**/
else if(umsg==WM_CHAR){
  WM_func_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_CHAR){
  WM_funcIME_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_STARTCOMPOSITION){
  WM_funcIME_STARTCOMPOSITION();
```

```
}/**else if(umsg)**/
else if(umsg==WM_IME_COMPOSITION){
  WM_funcIME_COMPOSITION(lparam);
}/**else if(umsg)**/
else if(umsg==WM_IME_ENDCOMPOSITION){
  WM_funcIME_ENDCOMPOSITION();
}/**else if(umsg)**/
else if(umsg==WM_CLOSE){
imm_close();
breaks(0);

if(dialogflag>0){
dialogflag=3;refill=0;
}
else if(menuflag>0){
menuflag=3;refill=0;
}
else{
refill=0;charflag=0;charcode=2;
}

return 1;
}/**else if(umsg)**/
else if(umsg==WM_PAINT){
restore_in_PAINT();

return 1;
}/**else if(umsg)**/
else{}

return 0;
}/** wndproc_deleted **/


int wndproc_BL(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
char gotoflag;

if(umsg==WM_KEYDOWN){
BitBltflag=0;BitBltflag_=0;

/************************* menu keydowns -> ****************************/

if(menuflag>0){

imm_pause();
```

```
if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_menu;}

gotoflag=1;

if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){  /* added */
  menuflag=3;refill=0;BitBltflag_=2;}
else if(GKS(VK_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(BitBltflag_==0) {BitBlt_menu();csr();}
else if(BitBltflag_==1) csr();
else{}

return 1;
}/**if(menuflag)**/

/************************** <- menu keydowns ****************************/

/************************** dialog keydowns -> ****************************/

if(dialogflag>0){

if(immflag==2) immflag=0;
if(usflag==1) usflag=0;

if(compflag) return 1;

if(cqflag==2){
  BitBltflag_=2;
  goto end_dialog;}
if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_dialog;}

gotoflag=1;
```

```
if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){  /* added */
  dialogflag=3;refill=0;BitBltflag_=2;
  /*if(GKS_(VK_SHIFT)<0) nocloseflag=1;
  else passflag=1;*/}
else if(GKS(VK_RETURN)<0){
  trim_dialog();
  if(GKS_(/*VK_CONTROL*/VK_SHIFT)<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(BitBltflag_==0) {BitBlt_dialog();csr();}
else if(BitBltflag_==1)             csr();
else{}

return 1;
}/**if(dialogflag)**/

/************************** <- dialog keydowns ****************************/

if(function==2){
imm_pause();
keydowns_f2();

return 1;
}

if(immflag==2) immflag=0;
if(usflag==1) usflag=0;
if(function==3) function=0;
if(function==4) function=1;

if(compflag) return 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){
  if(fn!=FMAX-1) dlgproc_FILE(0);}
else if(GKS('A')<0){
  if(fn!=FMAX-1) dlgproc_FILE(1);}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==4*/gotoflag==-1){
  if(GKS(VK_F12)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F1)<0) cqflag=0;  /* Esc+ */
  BitBltflag=/*2*/1;
  goto end;}

if(gotoflag==1) goto end;else gotoflag=1;

/*9*/
if(GKS(VK_F2)<0){
  if(fn!=FMAX-1){
  if(GKS_(VK_CONTROL)<0) u_s_flag=1;else u_s_flag=0;
  dlgproc_JUMP();}}

else if(GKS(VK_RETURN)<0){
  if(Enter(0)==1) goto end;}

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('I')<0){
  cqflag=1;prompt_cq(0);}
else if(GKS(VK_ESCAPE)<0){
  if(fn!=FMAX-1) {cqflag=3;prompt_cq(1);}}
else if(GKS(VK_F12)<0){  /* added */
  if(fn!=FMAX-1) {cqflag=3;prompt_cq(1);cqflag++;}}
else if(GKS_(VK_CONTROL)<0 && GKS('Q')<0){
  cqflag=5;prompt_cq(2);}

else if(GKS_(VK_CONTROL)<0 && GKS('Y')<0){
  YKP(0);}
else if(GKS_(VK_CONTROL)<0 && GKS('K')<0){
  YKP(1);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('M')<0){  /* added */
  YKP(1);
  charflag=0;charcode=2;BitBltflag=/*2*/1;
```

```
  Quick_Find(2,1);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('N')<0){  /* added */
  YKP(1);
  charflag=0;charcode=2;BitBltflag=/*2*/1;
  Quick_Find(2,2);}
else if(GKS_(VK_CONTROL)<0 && GKS('2')<0){
  if(GKS_(VK_MENU)<0) FILE_jump(1);else FILE_jump(0);}
else if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)<0 && GKS('8')<0){  /* to zzz.string */
  if(fn!=FMAX-1) FILE_ref_tmp(1);}
else if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)<0 && GKS('9')<0){  /* to zzz.find */
  if(fn!=FMAX-1) FILE_ref_tmp(0);}

else if(GKS(VK_F6)<0){
  paste=0;cut=0;/*monitorline(1);BitBltflag=2;*/}
else if(GKS_(VK_SHIFT)<0 && GKS(VK_F7)<0){
  paste=2;cut=0;/*monitorline(1);BitBltflag=2;*/}
else if(GKS(VK_F7)<0){
  paste=1;cut=0;/*monitorline(1);BitBltflag=2;*/}

else if(GKS(VK_F8)<0){
  cut=0;/*monitorline(1);BitBltflag=2;*/}

else if(GKS(VK_F5)<0 && refflag==0){
  Replace();}
else if(GKS(VK_F9)<0 && refflag==0){
  Find(2);}

else {gotoflag=0;}

if(gotoflag==1) goto end;

end_left:
gotoflag=left_keydowns();

if(gotoflag==0) goto end_;  /* no job */

end:
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(umsg)**/
else if(umsg==WM_SYSKEYDOWN){
if(cqflag) {extraline(1);cqflag=0;imm_restart();}
}/**else if(umsg)**/
else if(umsg==WM_CHAR){
  WM_func_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_CHAR){
  WM_funcIME_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_STARTCOMPOSITION){
  WM_funcIME_STARTCOMPOSITION();
}/**else if(umsg)**/
else if(umsg==WM_IME_COMPOSITION){
  WM_funcIME_COMPOSITION(lparam);
}/**else if(umsg)**/
else if(umsg==WM_IME_ENDCOMPOSITION){
  WM_funcIME_ENDCOMPOSITION();
}/**else if(umsg)**/
else if(umsg==WM_CLOSE){
imm_close();

if(dialogflag>0){
dialogflag=3;refill=0;
breaks(0);
}
else if(menuflag>0){
menuflag=3;refill=0;
breaks(0);
}
else{
cut=0;/*monitorline(1);BitBltflag=2;*//*beep(50);*/
/*if(BitBltflag==0) {*/BitBlt_full();csr();/*}*/
breaks(1);
}

return 1;
}/**else if(umsg)**/
else if(umsg==WM_PAINT){
restore_in_PAINT();

return 1;
}/**else if(umsg)**/
```

```
else{}

return 0;
}/** wndproc_BL **/


int wndproc(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
char gotoflag;

if(umsg==WM_KEYDOWN){
BitBltflag=0;BitBltflag_=0;

/************************* menu keydowns -> ****************************/

if(menuflag>0){

imm_pause();

if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_menu;}

gotoflag=1;

if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){  /* added */
  menuflag=3;refill=0;BitBltflag_=2;}
else if(GKS(VK_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(BitBltflag_==0) {BitBlt_menu();csr();}
else if(BitBltflag_==1) csr();
else{}

return 1;
}/**if(menuflag)**/

/************************* <- menu keydowns ****************************/
```

```
/************************* dialog keydowns -> ****************************/

if(dialogflag>0){

if(immflag==2) immflag=0;
if(usflag==1) usflag=0;

if(compflag) return 1;

if(cqflag==2){
  BitBltflag_=2;
  goto end_dialog;}
if(cqflag==6){
  /*BitBltflag_=2;*/
  goto end_left_dialog;}

gotoflag=1;

if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F12)<0 || GKS(VK_F1)<0){  /* added */
  dialogflag=3;refill=0;BitBltflag_=2;
  if(GKS_(VK_SHIFT)<0) nocloseflag=1;
  else passflag=1;}
else if(GKS(VK_RETURN)<0){
  trim_dialog();
  if(GKS_(/*VK_CONTROL*/VK_SHIFT)<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(BitBltflag_==0) {BitBlt_dialog();csr();}
else if(BitBltflag_==1)             csr();
else{}

return 1;
}/**if(dialogflag)**/

/************************* <- dialog keydowns ****************************/

if(function==2){
```

```
imm_pause();
keydowns_f2();

return 1;
}

if(immflag==2) immflag=0;
if(usflag==1) usflag=0;
if(function==3) function=0;
if(function==4) function=1;

if(compflag) return 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){
  open_file(0);}
else if(GKS('M')<0){
  open_file(1);}
else if(GKS('N')<0){
  open_file(2);}
else if(GKS('Q')<0 || GKS('U')<0){
  close_all();}
else if(GKS('W')<0 || GKS('Y')<0){
  if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)<0) end_ble();
  else save_all(1);}
else if(GKS('X')<0 || GKS('V')<0){
  if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)<0) close_open(0);
  else close_open(1);}
else if(GKS('C')<0){
  close_file();}
else if(GKS('E')<0 || GKS('T')<0){
  save_all(0);}
else if(GKS('S')<0 || GKS('G')<0){
  dlgproc_SAVE(0);}
```

```
else if(GKS('A')<0){
  dlgproc_SAVE(1);}
else if(GKS('R')<0){
  dlgproc_REN();}
else if(GKS('I')<0){
  dlgproc_INS();}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==4*/gotoflag==-1){
  if(GKS(VK_F12)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F1)<0) cqflag=0;  /* Esc+ */
  BitBltflag=/*2*/1;
  goto end;}

if(gotoflag==1) goto end;else gotoflag=1;

/*9*/
if(GKS_(VK_SHIFT)<0 && GKS(VK_F1)<0){
  if(GKS_(VK_CONTROL)<0) copy_cfg();
  else use_selector(4);}
else if(GKS(VK_F2)<0){
  if(GKS_(VK_CONTROL)<0) u_s_flag=1;else u_s_flag=0;
  dlgproc_JUMP();}
else if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)<0 && GKS(VK_UP)<0){
  switch_division(0);}
else if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)<0 && GKS(VK_DOWN)<0){
  switch_division(1);}
else if(GKS_(VK_SHIFT)<0 && GKS(VK_F3)<0){
  restore_display();}
else if(GKS(VK_F3)<0){
  divide_display();}
else if((GKS_(VK_SHIFT)<0 && GKS(VK_F4)<0) || (GKS_(VK_SHIFT)<0 && GKS(VK_F11)<0)){
  if(GKS_(VK_CONTROL)<0) systemflag=1;else systemflag=0;
  use_selector(3);}
else if(GKS(VK_F4)<0|| GKS(VK_F11)<0){
  show_file();}

else if(GKS_(VK_CONTROL)>=0 && GKS_(VK_SHIFT)<0 && GKS(VK_F8)<0){
  use_deleted(0);}
else if(GKS_(VK_CONTROL)>=0 && GKS_(VK_SHIFT)>=0 && GKS(VK_F8)<0){
  use_selector(/*1*/6);}

else if(GKS(VK_RETURN)<0){
  if(Enter(0)==1) goto end;}
```

```
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('I')<0){
  cqflag=1;prompt_cq(0);}
else if(GKS(VK_ESCAPE)<0){
  cqflag=3;prompt_cq(1);}
else if(GKS(VK_F12)<0){  /* added */
  cqflag=3;prompt_cq(1);cqflag++;}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)<0 && GKS('Q')<0){
  cqflag=5;prompt_cq(2);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('Q')<0){
  cqflag=7;prompt_cq(3);}

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('Y')<0){
  YKP_word(0);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('K')<0){
  YKP_word(1);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('M')<0){  /* added */
  YKP_word(1);
  charflag=0;charcode=2;BitBltflag=/*2*/1;
  Quick_Find(1,1);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('N')<0){  /* added */
  YKP_word(1);
  charflag=0;charcode=2;BitBltflag=/*2*/1;
  Quick_Find(1,2);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && (GKS('P')<0 || GKS('O')<0)){
  if(GKS('O')<0) MOVEcsr*=-1;YKP_word(2);if(GKS('O')<0) MOVEcsr*=-1;}
else if(GKS_(VK_CONTROL)<0 && GKS('Y')<0){
  YKP(0);}
else if(GKS_(VK_CONTROL)<0 && GKS('K')<0){
  YKP(1);}
else if(GKS_(VK_CONTROL)<0 && (GKS('P')<0 || GKS('O')<0)){
  if(GKS('O')<0) MOVEcsr*=-1;YKP(2);if(GKS('O')<0) MOVEcsr*=-1;}
else if(GKS_(VK_CONTROL)<0 && GKS('2')<0){
  if(GKS_(VK_MENU)<0) FILE_jump(1);else FILE_jump(0);}
else if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)<0 && GKS('8')<0){  /* to zzz.string */
  FILE_ref_tmp(1);}
else if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)<0 && GKS('9')<0){  /* to zzz.find */
  FILE_ref_tmp(0);}

else if(GKS(VK_F6)<0){
  paste=0;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS_(VK_SHIFT)<0 && GKS(VK_F7)<0){
  paste=2;cut=0;monitorline(1);BitBltflag=2;}
else if(GKS(VK_F7)<0){
  paste=1;cut=0;monitorline(1);BitBltflag=2;}
```

```
else if(GKS(VK_F5)<0){
  Replace();}
else if(GKS(VK_F9)<0){
  Find(1);}

else {gotoflag=0;}

if(gotoflag==1) goto end;

end_left:
gotoflag=left_keydowns();

if(gotoflag==0) goto end_;  /* no job */

end:
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(umsg)**/
else if(umsg==WM_SYSKEYDOWN){
if(cqflag) {/*beep(50);*/extraline(1);cqflag=0;imm_restart();}
}/**else if(umsg)**/
else if(umsg==WM_CHAR){
  WM_func_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_CHAR){
  WM_funcIME_CHAR(wparam);
return 1;
}/**else if(umsg)**/
else if(umsg==WM_IME_STARTCOMPOSITION){
  WM_funcIME_STARTCOMPOSITION();
}/**else if(umsg)**/
else if(umsg==WM_IME_COMPOSITION){
  WM_funcIME_COMPOSITION(lparam);
}/**else if(umsg)**/
else if(umsg==WM_IME_ENDCOMPOSITION){
  WM_funcIME_ENDCOMPOSITION();
}/**else if(umsg)**/
```

```
else if(umsg==WM_CLOSE){
imm_close();

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(umsg)**/
else if(umsg==WM_PAINT){
restore_in_PAINT();

return 1;
}/**else if(umsg)**/
else{}

return 0;
}/** wndproc **/


void WM_func_CHAR(WPARAM wparam)
{
unsigned char charcode_tmp;

BitBltflag=0;BitBltflag_=0;

charcode_tmp=(unsigned char)wparam;    /* bridge */
/*if(charcode_tmp<0x20 || GKS_(VK_CONTROL)<0 || GKS_(VK_MENU)<0){*/
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(driveflag) 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();
}
else {BitBltflag_=2;}

if(BitBltflag_==0) {BitBlt_menu();csr();}
else if(BitBltflag_==1)           csr();
else{}

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);
}

if(BitBltflag_==0) {BitBlt_dialog();csr();}
else if(BitBltflag_==1)             csr();
else{}
```

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

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


void InputPosition(HIMC himc,int icsr,int jcsr)
{
int dx,dy;

myime.dwStyle=CFS_POINT;

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+DSHIFT_2;

myime.ptCurrentPos=point;
```

```
ImmSetCompositionWindow(himc,&myime);
}/** InputPosition **/


void WM_funcIME_CHAR(WPARAM wparam)
{
char flag_,function_old;
long k;
unsigned char db[2];

if(dialogflag>0){
if(HIBYTE(wparam)){
if(dbflag){
db[0]=HIBYTE(wparam);
db[1]=LOBYTE(wparam);

tailcheck_dialog();

flag_=0;

kmax_dialog+=2;
if(kmax_dialog>ASIZEM-1) {beep(50);kmax_dialog-=2;flag_=1;}
else{
k=firstk_dialog+icsr;
/*memcpy(&p_dialog[k+2],&p_dialog[k],kmax_dialog-2-k+1);*/
memcpy_(&p_dialog[0],k+2,&p_dialog[0],k,kmax_dialog-2-k+1);
/*memcpy(&p_dialog[k],&db[0],2);*/
memcpy_(&p_dialog[0],k,&db[0],0,2);
}

/*page_firstk_dialog(firstk_dialog);*/

if(flag_==0){
csr_right_dialog();
}
else{
dbcount=dbsize;
}
}/**if(dbflag)**/

dbcount+=2;
}/**if(HIBYTE)**/
else{
if(dbflag){
if(insertion_dialog(LOBYTE(wparam))==1) dbcount=dbsize;
if(!compflag) dbsize=1;             /* hankaku space */
```

```
}/**if(dbflag)**/

dbcount+=1;
}/**else(HIBYTE)**/

if(dbflag==1 && dbcount>=dbsize){   /* > : notice ! */
dbflag=0;
page_firstk_dialog(firstk_dialog);csr();       /* csr() : for Windows 2000 */

if(compflag){
/*myime.dwStyle=CFS_POINT;
point.x=(icsr+DI_d)*UDX;point.y=(jcsr+DJ_d)*UDY+DSHIFT_2;
myime.ptCurrentPos=point;
ImmSetCompositionWindow(himc,&myime);*/
InputPosition(himc,icsr,jcsr);
}
}

return;
}/**if(dialogflag)*****************************/

if(function>=2) return;           /* 3,4 : for Windows ? */
if(filerflag) return;

if(HIBYTE(wparam)){
stock_db[dbcount]=HIBYTE(wparam);
stock_db[dbcount+1]=LOBYTE(wparam);

dbcount+=2;
}/**if(HIBYTE)**/
else{
stock_db[dbcount]=LOBYTE(wparam);
if(!compflag) dbsize=1;             /* hankaku space */

dbcount+=1;
}/**else(HIBYTE)**/

if(dbcount==dbsize){
if(compflag==0 && dbsize==1){
dbflag=0;
insertion(stock_db[0]);csr();       /* csr() : for Windows 2000 */
}/**if(compflag,dbsize)**/
else{
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;
}
dbflag=0;
page_firstk(firstk);csr();       /* csr() : for Windows 2000 */

if(flag_==0){
/*csr_right();monitorline(1);*/
if(editflag[fn]>-1) editflag[fn]=1;else editflag[fn]=-2;
}
}/**else(compflag,dbsize)**/

if(compflag){
/*myime.dwStyle=CFS_POINT;
point.x=(icsr+DI)*UDX;point.y=(jcsr+DJ)*UDY+DSHIFT_2;
myime.ptCurrentPos=point;
ImmSetCompositionWindow(himc,&myime);*/
InputPosition(himc,icsr,jcsr);
}

if(fn!=FMAX-1 && ftp>0) write_3vals(ftp-1);
}/**if(dbcount)**/
}/** WM_funcIME_CHAR **/



void WM_funcIME_STARTCOMPOSITION(void)
{
/*beep(50);delay_(100);beep(50);delay_(100);*/

if(immflag==0){
compflag=1;

himc=ImmGetContext(hwnd);
ImmGetCompositionFont(himc,&myimefont);
myimefont.lfHeight=UDY;
myimefont.lfWidth=UDX;
strcpy(myimefont.lfFaceName,"MSMINCHO");
ImmSetCompositionFont(himc,&myimefont);
```

```
}

/*return 1;*/
}/** WM_funcIME_STARTCOMPOSITION **/


void WM_funcIME_COMPOSITION(LPARAM lparam)
{
static unsigned char dbbuf[ASIZE]={0};

/*beep(500);delay_(100);*/

if(lparam & GCS_RESULTSTR){
if(compflag) dbsize=ImmGetCompositionString(himc,GCS_RESULTSTR,dbbuf,sizeof(dbbuf));
else dbsize=2;                        /* space */
dbflag=1;
dbcount=0;
}
else{
/*myime.dwStyle=CFS_POINT;
if(dialogflag>0) {point.x=(icsr+DI_d)*UDX;point.y=(jcsr+DJ_d)*UDY+DSHIFT_2;}
else {point.x=(icsr+DI)*UDX;point.y=(jcsr+DJ)*UDY+DSHIFT_2;}
myime.ptCurrentPos=point;
ImmSetCompositionWindow(himc,&myime);*/
InputPosition(himc,icsr,jcsr);
}

/*return 1;*/
}/** WM_funcIME_COMPOSITION **/


void WM_funcIME_ENDCOMPOSITION(void)
{
/*beep(50);*/

if(cqflag) {/*beep(50);*/extraline(1);cqflag=0;imm_restart();}
if(puts_mline_flag) {puts_mline_flag=0;extraline(1);}

if(compflag==0) {if(immflag!=1) csr();imeendflag=0;}
else imeendflag=1;
compflag=0;
dbflag=0;
/*immflag=0;*/                          /* no problem ? */

ImmReleaseContext(hwnd,himc);
}/** WM_funcIME_ENDCOMPOSITION **/
```

```
char left_keydowns(void)
{
char gotoflag,flag_;
long k;
unsigned char db[2];

gotoflag=1;

if(cqflag==8){                      /* ~^Q-> */
if(GKS('E')<0){
  switch_division(0);}
else if(GKS('X')<0){
  switch_division(1);}
else if(GKS('V')<0){
  move_and_paste();}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==8*/gotoflag==-1){
  if(GKS(VK_F12)<0 || GKS(VK_PAUSE)<0 || GKS(VK_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 && cut==0 && filerflag==0 &&
        refflag==0 && deletedflag==0){
  file_attri();}

else if(GKS('H')<0 && cut==0 && filerflag==0){
  half_word(0);}
else if(GKS('G')<0 && cut==0 && filerflag==0){
  half_word(1);}
else if(GKS('Y')<0 && cut==0 && filerflag==0){
```

```
  half_line(0);}
else if(GKS('T')<0 && cut==0 && filerflag==0){
  half_line(1);}

else if(GKS('E')<0){
  csr_column_home();}
else if(GKS('X')<0){
  csr_column_end();}
else if(GKS('S')<0 && filerflag==0){
  csr_row_home();}
else if(GKS('D')<0 && filerflag==0){
  csr_row_end();}

else if(GKS('F')<0 && filerflag==0){
  find_0x1a(0);}
else if(GKS('A')<0 && filerflag==0){
  find_0x1a(1);}

else if(GKS('R')<0){
  text_home();}
else if(GKS('C')<0){
  text_end();}

else if(GKS('W')<0){
  page_up();}
else if(GKS('Z')<0){
  page_down();}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==6*/gotoflag==-1){
  if(GKS(VK_F12)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F1)<0) cqflag=0;  /* Esc+ */
  BitBltflag=/*2*/1;
  goto end_lk;}

if(gotoflag==1) goto end_lk;else gotoflag=1;

if(GKS_(VK_CONTROL)<0 && GKS('B')<0 && filerflag==0){
  cut=1;BL();}
else if(GKS_(VK_CONTROL)<0 && GKS('L')<0 && filerflag==0){
  cut=2;BL();}

else if(GKS_(VK_SHIFT)<0 && GKS(VK_DELETE)<0 && filerflag==0){
  half_word(0);}
```

```
else if(GKS_(VK_SHIFT)<0 && GKS(VK_BACK)<0 && filerflag==0){
#if NOTEKBD==0
  half_word(1);
#else
  delorbs=0;
  deletion();
#endif
  }
else if(GKS_(VK_CONTROL)<0 && GKS(VK_DELETE)<0 && filerflag==0){
  half_line(0);}
else if(GKS_(VK_CONTROL)<0 && GKS(VK_BACK)<0 && filerflag==0){
  half_line(1);}

else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)>=0 && GKS(VK_DELETE)<0 && filerflag==0){
  delorbs=0;
  deletion();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)>=0 && GKS(VK_BACK)<0 && filerflag==0){
  delorbs=1;
  backspace();}
else if(GKS_(VK_CONTROL)<0 && GKS('U')<0 && filerflag==0){
  overwrite();
  insertion_u();}

else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)>=0 && GKS(VK_UP)<0){
  csr_column_home();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)>=0 &&
        GKS(VK_DOWN)<0){
  csr_column_end();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)>=0 && GKS(VK_HOME)<0 && filerflag==0){
  csr_row_home();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)>=0 && GKS(VK_END)<0 && filerflag==0){
  csr_row_end();}

else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)>=0 && GKS(VK_UP)<0){
  csr_up();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)>=0 && GKS(VK_DOWN)<0){
  csr_down();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)>=0 && GKS(VK_LEFT)<0 && filerflag==0){
  csr_left();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)>=0 && GKS(VK_RIGHT)<0 && filerflag==0){
  csr_right();}

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)>=0 && GKS(VK_PRIOR)<0){
  scroll_up(0);}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)>=0 && GKS(VK_NEXT)<0){
  scroll_down(0);}
```

```
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)<0 && GKS(VK_HOME)<0){
  centering_csr();}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)>=0 && GKS(VK_HOME)<0){
  centering_theline();}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS(VK_HOME)<0){
  page_firstk(topp[jcsr]);
  jcsr=0;/*csr();*/
  }

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('T')<0){
  page_firstk(topp[jcsr]);
  jcsr=0;/*csr();*/
  }
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('V')<0){
  centering_theline();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)<0 && GKS('V')<0){
  centering_csr();}

else if((GKS_(VK_CONTROL)<0 && GKS(VK_PRIOR)<0) ||
        (GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS(VK_UP)<0)){
  text_home();}
else if((GKS_(VK_CONTROL)<0 && GKS(VK_NEXT)<0) ||
        (GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS(VK_DOWN)<0)){
  text_end();}

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)>=0 && GKS(VK_UP)<0){
  page_up();}
else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)>=0 && GKS(VK_DOWN)<0){
  page_down();}

else if(GKS(VK_PRIOR)<0){
  if(function==0) page_up();
  else{
  charflag=0;charcode=0;
  BitBltflag=2;}}
else if(GKS(VK_NEXT)<0){
  if(function==0) page_down();
  else{
  charflag=0;charcode=1;
  BitBltflag=2;}}

else if(GKS_(VK_CONTROL)<0 && GKS(VK_RIGHT)<0 && filerflag==0){
  find_0x1a(0);}
else if(GKS_(VK_CONTROL)<0 && GKS(VK_LEFT)<0 && filerflag==0){
  find_0x1a(1);}
else if(GKS_(VK_SHIFT)<0 && GKS(VK_RIGHT)<0 && filerflag==0){
```

```
  find_word(0);}
else if(GKS_(VK_SHIFT)<0 && GKS(VK_LEFT)<0 && filerflag==0){
  find_word(1);}

else if(GKS_(VK_SHIFT)<0 && GKS(VK_INSERT)<0 && refflag==0){
  charflag=0;charcode=2;
  BitBltflag=2;}
else if((GKS(VK_PAUSE)<0 || (GKS_(VK_SHIFT)>=0 && GKS(VK_F1)<0)) && refflag==0){
  charflag=0;charcode=2;
  BitBltflag=2;}
else if(GKS_(VK_CONTROL)<0 && GKS(VK_INSERT)<0 && filerflag==0){
  if(insorover==0) insorover=1;else insorover=0;
  extraline(1);BitBltflag=2;}
else if(GKS(VK_TAB)<0 && filerflag==0){
  overwrite();
  insertion(0x09);}

else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)<0 && GKS('G')<0 && filerflag==0){
  delorbs=0;
  deletion();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)<0 && GKS('H')<0 && filerflag==0){
  delorbs=1;
  backspace();}

else if(GKS_(VK_CONTROL)<0 && GKS('E')<0){
  csr_up();}
else if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)>=0 && GKS('X')<0){
  csr_down();}
else if(GKS_(VK_CONTROL)<0 && GKS('S')<0 && filerflag==0){
  csr_left();}
else if(GKS_(VK_CONTROL)<0 && GKS('D')<0 && filerflag==0){
  csr_right();}

else if(GKS_(VK_CONTROL)<0 && GKS('F')<0 && filerflag==0){
  find_word(0);}
else if(GKS_(VK_CONTROL)<0 && GKS('A')<0 && filerflag==0){
  find_word(1);}

else if(GKS_(VK_CONTROL)<0 && GKS('R')<0){
  if(function==0) page_up();
  else{
  charflag=0;charcode=0;
  BitBltflag=2;}}
else if(GKS_(VK_CONTROL)<0 && GKS('C')<0){
  if(function==0) page_down();
  else{
```

```
  charflag=0;charcode=1;
  BitBltflag=2;}}

else if(GKS_(VK_CONTROL)<0 && GKS('9')<0 && refflag==0){
  Quick_Find(0,0);}

else if(GKS_(VK_CONTROL)<0 && GKS_(VK_MENU)>=0 && GKS('W')<0){
  scroll_up(0);}
else if(GKS_(VK_CONTROL)<0 && GKS('Z')<0){
  scroll_down(0);}

else if(GKS_(VK_CONTROL)<0 && GKS('M')<0){
  if(refflag){
  if(GKS_(VK_SHIFT)<0/* || GKS_(VK_CONTROL)<0*/){
  /*if(GKS_(VK_SHIFT)<0) {*/uflag=1;csr_row_home();/*}*/

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

  /*if(GKS_(VK_SHIFT)<0) */uflag=0;}
  else{
  refill=-2;charflag=0;charcode=2;
  BitBltflag=2;}}/**if(refflag)**/
  else if(filerflag){
  refill=-2;if(GKS_(VK_SHIFT)<0) refill--;charflag=0;charcode=2;
  BitBltflag=2;}/**else if(filerflag)**/
  else{
  if(GKS_(VK_SHIFT)<0) {uflag=1;csr_row_home();}
  if(AINDENT==1) /*{if(GKS_(VK_CONTROL)>=0) */lumpflag=1;/*}*/
  /*else {if(GKS_(VK_CONTROL)<0) lumpflag=1;}*/

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

  if(AINDENT==1) /*{if(GKS_(VK_CONTROL)>=0) */autoindent();/*}*/
  /*else {if(GKS_(VK_CONTROL)<0) autoindent();}*/
  if(GKS_(VK_SHIFT)<0) uflag=0;}/**else(refflag,filerflag)**/}
else if(GKS_(VK_CONTROL)<0 && GKS('N')<0 && filerflag==0){
  if(refflag){
  /*if(GKS_(VK_SHIFT)<0) {*/uflag=1;csr_row_home();/*}*/

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

  /*if(GKS_(VK_SHIFT)<0) */uflag=0;}/**if(refflag)**/
  else{
  /*if(GKS_(VK_SHIFT)<0) {*/uflag=1;csr_row_home();/*}*/
  if(AINDENT==1) /*{if(GKS_(VK_CONTROL)>=0) */lumpflag=1;/*}*/
  /*else {if(GKS_(VK_CONTROL)<0) lumpflag=1;}*/
```

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

  if(AINDENT==1) /*{if(GKS_(VK_CONTROL)>=0) */autoindent();/*}*/
  /*else {if(GKS_(VK_CONTROL)<0) autoindent();}*/
  /*if(GKS_(VK_SHIFT)<0) */uflag=0;}/**else(refflag)**/}

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 &&
        GKS('H')<0 && filerflag==0){
  lumpflag=1;

  himc_=ImmGetContext(hwnd);
  if(ImmGetOpenStatus(himc_)==FALSE){
  ImmReleaseContext(hwnd,himc_);
  flag_=0;
  uflag=1;
  if(insertion(' ')==1) {lumpflag=0;uflag=0;goto end_lk;}
  uflag=0;
  }/**if(ImmGetOpenStatus())**/
  else{
  ImmReleaseContext(hwnd,himc_);
  tailcheck();

  db[0]=/*0x81*/SPC1;
  db[1]=/*0x40*/SPC2;

  k=top_icsr(/*firstline+*/jcsr,icsr);
  flag_=pdata_increase(k,&db[0],2);

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

  if(flag_==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){
  csr_row_home_dialog();}
else if(GKS('D')<0){
  csr_row_end_dialog();}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==6*/gotoflag==-1){
  if(GKS(VK_F12)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F1)<0) cqflag=0;  /* Esc+ */
  /*BitBltflag_=2;*/
  goto end_lk_dialog;}

if(gotoflag==1) goto end_lk_dialog;else gotoflag=1;

if(GKS(VK_DELETE)<0){
  deletion_dialog();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)>=0 && GKS(VK_BACK)<0){
  backspace_dialog();}

else if(GKS(VK_UP)<0){
  restore_dialog();}
else if(GKS(VK_DOWN)<0){
  clear_dialog(1);}
else if(GKS_(VK_CONTROL)>=0 && GKS(VK_LEFT)<0){
  csr_left_dialog();}
else if(GKS_(VK_CONTROL)>=0 && GKS(VK_RIGHT)<0){
  csr_right_dialog();}

else if(GKS(VK_HOME)<0 || (GKS_(VK_CONTROL)<0 && GKS(VK_LEFT)<0)){
  csr_row_home_dialog();}
else if(GKS(VK_END)<0 || (GKS_(VK_CONTROL)<0 && GKS(VK_RIGHT)<0)){
  csr_row_end_dialog();}

else if(GKS(VK_PRIOR)<0){
  page_up_dialog();}
else if(GKS(VK_NEXT)<0){
  page_down_dialog();}
```

```
else if(GKS_(VK_CONTROL)<0 && GKS(VK_INSERT)<0){
  if(insorover==0) insorover=1;else insorover=0;
  extraline(1);BitBltflag_=2;}

else if(GKS_(VK_SHIFT)<0 && GKS_(VK_CONTROL)<0 && GKS('I')<0){
  cqflag=1;prompt_cq(0);}
else if(GKS_(VK_CONTROL)<0 && GKS('Q')<0){
  cqflag=5;prompt_cq(2);}
else if(GKS(VK_TAB)<0){
  overwrite();
  insertion_dialog(0x09);}

else if(GKS_(VK_CONTROL)<0 && GKS('G')<0){
  deletion_dialog();}
else if(GKS_(VK_SHIFT)>=0 && GKS_(VK_CONTROL)<0 && GKS('H')<0){
  backspace_dialog();}

else if(GKS_(VK_CONTROL)<0 && GKS('E')<0){
  restore_dialog();}
else if(GKS_(VK_CONTROL)<0 && GKS('X')<0){
  clear_dialog(1);}
else if(GKS_(VK_CONTROL)<0 && GKS('S')<0){
  csr_left_dialog();}
else if(GKS_(VK_CONTROL)<0 && GKS('D')<0){
  csr_right_dialog();}

else if(GKS_(VK_CONTROL)<0 && GKS('R')<0){
  page_up_dialog();}
else if(GKS_(VK_CONTROL)<0 && GKS('C')<0){
  page_down_dialog();}

else if(GKS_(VK_CONTROL)<0 && GKS('M')<0){
  trim_dialog();
  if(GKS_(/*VK_CONTROL*/VK_SHIFT)<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){
  csr_column_home_menu();}
else if(GKS('X')<0){
  csr_column_end_menu();}

else {gotoflag=-1;}
}/**if(cqflag)**/
else {gotoflag=0;}

if(/*cqflag==6*/gotoflag==-1){
  if(GKS(VK_F12)<0 || GKS(VK_PAUSE)<0 || GKS(VK_F1)<0) cqflag=0;  /* Esc+ */
  /*BitBltflag_=2;*/
  goto end_lk_menu;}

if(gotoflag==1) goto end_lk_menu;else gotoflag=1;

if(GKS_(VK_CONTROL)<0 && GKS(VK_UP)<0){
  csr_column_home_menu();}
else if(GKS_(VK_CONTROL)<0 && GKS(VK_DOWN)<0){
  csr_column_end_menu();}

else if(GKS(VK_UP)<0){
  csr_up_menu();}
else if(GKS(VK_DOWN)<0){
  csr_down_menu();}

else if(GKS_(VK_CONTROL)<0 && GKS('Q')<0){
  cqflag=5;prompt_cq(2);}

else if(GKS_(VK_CONTROL)<0 && GKS('E')<0){
  csr_up_menu();}
else if(GKS_(VK_CONTROL)<0 && GKS('X')<0){
  csr_down_menu();}

else if(GKS_(VK_CONTROL)<0 && GKS('M')<0){
  menuflag=2;refill=0;BitBltflag_=2;}

else {BitBltflag_=2;}

end_lk_menu:{}
}/** left_keydowns_menu **/
```

```
/* cd program                                              */
/* chdir_by_filer.c(cf.c)                                  */
/* by Morio Kikuchi 2018.1.1                               */
/* SYSTEM:DOS window, FreeDOS(DPMI needed)                 */
/* COMPILER:djgpp 2.05                                     */
/* COMMANDLINE:gcc -Dfar= -o cf.exe cf.c                   */
/* USAGE:1st:filer in fgrep.exe                            */
/*                     #define GRP_or_EDT 1                */
/*                     #define FF_2 2                      */
/*                     #define FF_2 3                      */
/*       2nd:cf.exe => dest directory by filer             */
/*       3rd:cf.exe => prev directory                      */
/*       4th:cf.exe => dest directory by filer             */

#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <unistd.h>
#include <dir.h>

#define ASIZE (MAXPATH+1)


int main(int argc,unsigned char **argv)
{
int i,length;
long fsize;
unsigned char home_global_GCD[ASIZE],home_global[ASIZE],home[ASIZE],
              dir[ASIZE];
FILE *fp;

getcwd((char *)home_global_GCD,ASIZE);
length=strlen(home_global_GCD);
/*printf(" %s\n",home_global_GCD);getch();*/

if(chdir("c:\\ble")==0){
strcpy(home_global,"c:\\ble\\");chdir((char *)home_global_GCD);
}
else if(chdir("d:\\ble")==0){
strcpy(home_global,"d:\\ble\\");chdir((char *)home_global_GCD);
}
else if(chdir("e:\\ble")==0){
strcpy(home_global,"e:\\ble\\");chdir((char *)home_global_GCD);
}
```

```
else{
strncpy(home_global,/*buf*/"c:\\",3);          /* c:\, d:\, e:\ */
home_global[3]='\0';
}

strcpy(home,home_global);
strcat(home,"cpage_f.bin");

if(access(home,0)==0 && access(home,4)==0){  /* e, r */
if((fp=fopen(home,"rb"))!=NULL){
fseek(fp,0,2);fsize=ftell(fp);

fseek(fp,0,0);
fread(dir,1,fsize,fp);
dir[fsize]='\0';

/*chdir(dir);*/

fclose(fp);
}
else goto end;



i=0;
while(1){
if(home_global_GCD[i]=='/') home_global_GCD[i]='\\';
i++;

if(i==length) break;
}



if(stricmp(home_global_GCD,dir)!=0){
strcpy(home,home_global);
strcat(home,"cpage_r.bin");

if((fp=fopen(home,"wb"))!=NULL){
fsize=strlen(home_global_GCD);
fwrite(home_global_GCD,1,fsize,fp);
fclose(fp);
}

chdir(dir);  /* filer() */
}/**if(stricmp)**/
```

```
else{
strcpy(home,home_global);
strcat(home,"cpage_r.bin");

if(access(home,0)==0 && access(home,4)==0){  /* e, r */
if((fp=fopen(home,"rb"))!=NULL){
fseek(fp,0,2);fsize=ftell(fp);

fseek(fp,0,0);
fread(dir,1,fsize,fp);
dir[fsize]='\0';

chdir(dir);  /* cd_filer.exe */

fclose(fp);
}
else goto end;
}/**if(access(r,0),access(r,4))**/
else goto end;
}/**else(stricmp)**/
}/**if(access(f,0),access(f,4))**/
else goto end;

end:{}
}/** main **/


@echo off
rem fgrep.exe
rem #define GRP_or_EDT 1
rem #define FF_2 0  /* LINEMODE => 0 */
rem #define FF_2 1  /* LINEMODE => 1 */
rem %1:filename, %2:linenumber
rem fgrep.exe %1 %2

rem fgrep_.bat
rem fgrep.exe
rem #define GRP_or_EDT 1
rem #define FF_2 2  /* LINEMODE => 0 */
rem #define FF_2 3  /* LINEMODE => 1 */
rem Use filer in fgrep.exe
fgrep.exe
cf.exe
```