Cellular Automaton Graphics(9)

Morio Kikuchi

Abstract：

Developing a regular polyhedron on a plane, setting discrete coordinates on the development and applying a boundary condition of regular polyhedron to it, we realize a symmetrical graphics.

1. Johnson solid

Figure 1 shows snub square antiprism which is one of Jhonson solids.

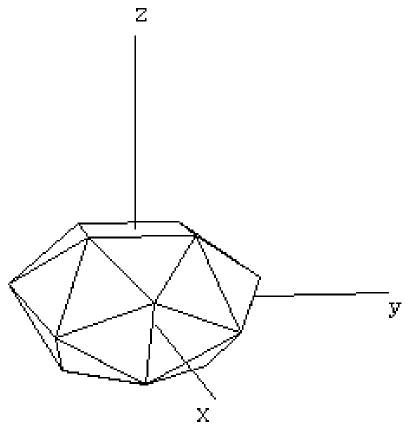
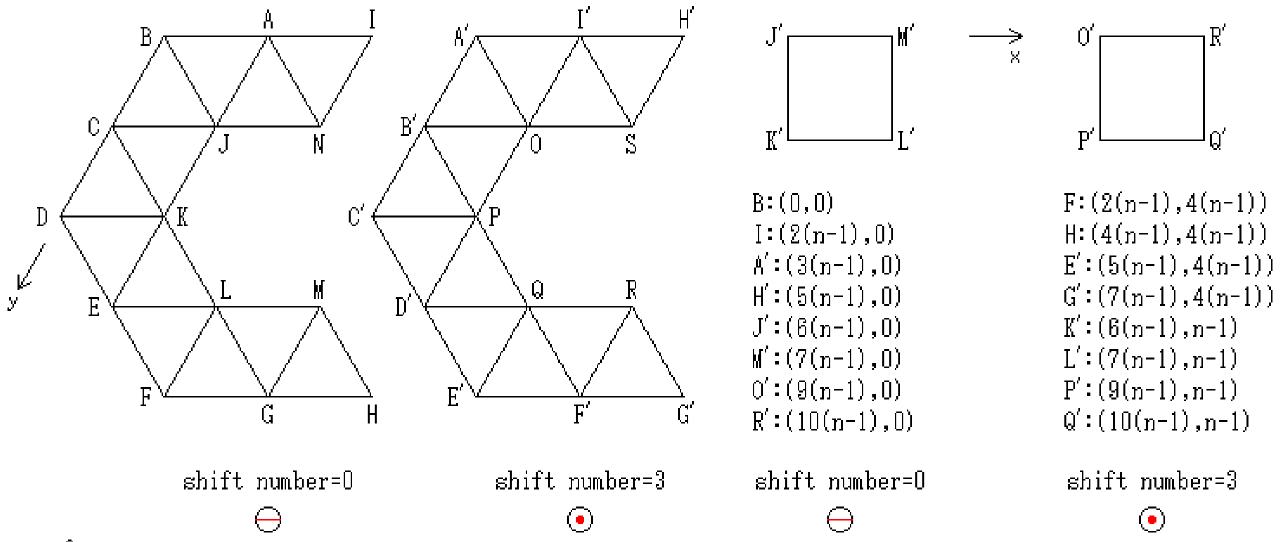


Figure 1

This figure has a shape that two bowls which consists of regular polygons are combined facing in opposite directions. The spatial angle difference of the two bowls is 45 °. Figure 2 is a plane expression of the figure.



B:(0,0)
I:(2(n-1),0)
A′:(3(n-1),0)
H′:(5(n-1),0)
J′:(6(n-1),0)
M′:(7(n-1),0)
O′:(9(n-1),0)
R′:(10(n-1),0)

F:(2(n-1),4(n-1))
H:(4(n-1),4(n-1))
E′:(5(n-1),4(n-1))
G′:(7(n-1),4(n-1))
K′:(6(n-1),n-1)
L′:(7(n-1),n-1)
P′:(9(n-1),n-1)
Q′:(10(n-1),n-1)

Figure 2

The shift number of the upper bowl is 0, the left figure which is C type is the second convergence face and the right figure which is square is the first convergence face. The shift number of the lower bowl is 3 and the second and first convergence faces are the same as the upper bowl. The second and first convergence faces are expressed by oblique and orthogonal coordinates respectively.

Correspondences of vertexes are expressed by prime.

・A-A′, B-B′, C-C′, D-D′, E-E′, F-F′, G-G′(H′), H(I)-I′
・J-J′, K-K′, L-L′, M(N)-M′, O-O′, P-P′, Q-Q′, R(S)-R′

Because the second convergence face is expressed being cut along one line segment, M and N、H and I、R and S、G′ and H′ express the correspondences of vertexes.

2. Symmetry

The symmetry axis of the figure in Figure 1 is z axis. We show examples of seed point arrangement below.

・2-fold symmetry：SP:2(upper 1st conv. face), SP:2(upper 1st conv. face)+2(lower 1st conv. face)
・2-fold symmetry：SP:2(upper 2nd conv. face)、SP:2(upper 2nd conv. face)+2(lower 2nd conv. face)
・4-fold symmetry：SP:4(upper 1st conv. face), SP:4(upper 1st conv. face)+4(lower 1st conv. face)
・4-fold symmetry：SP:4(upper 2nd conv. face)、SP:4(upper 2nd conv. face)+4(lower 2nd conv. face)

Because the spatial angle difference between the upper and lower first convergence faces is 45 °, a symmetrical graphics of SP:2+2 is not odd tunnel.

In Figure 2, we resolved the figure into the first and second convergence faces for convenience. In fact, the second convergence face can be resolved into the second and third convergence faces.

3. Neighborhood

We express examples of a neighborhood in which jump is done with angle.

・MH：120 °,180 °,240 °,300 °+120 °,180 °(NI)
・NI：60 °,120 °,180 °,240 °+180 °,240 °(MH)
・RG′：120 °,180 °,240 °,300 °+120 °,180 °(SH′)
・SH′：60 °,120 °,180 °,240 °+180 °,240 °(RG′)

・JK：60 °,120 °,180 °,240 °+0 °(J′K′)
・J′K′：-90 °,0 °,90 °+120 °,180 °(JK)
・OP：60 °,120 °,180 °,240 °+0 °(O′P′)
・O′P′：-90 °,0 °,90 °+120 °,180 °(OP)

・AB：180 °,240 °,300 °,360 °+0 °,-60 °(A′B′)
・A′B′：-120 °,-60 °,0 °,60 °+(-60) °,-120 °(AB)

・J：0 °,60 °,120 °,180 °,240 °
・J′：-90 °,0 °+60 °,120 °,180 °(J)
・O：0 °,60 °,120 °,180 °,240 °
・O′：-90 °,0 °+60 °,120 °,180 °(O)
・M：180 °,240 °,300 °+120 °,180 °(N)
・N：60 °,120 °,180 °+180 °,240 °(M)
・M′：180 °,270 °+240 °,300 °(M)+120 °(N)
・R：180 °,240 °,300 °+120 °,180 °(S)
・S：60 °,120 °,180 °+180 °,240 °(R)
・R′：180 °,270 °+240 °,300 °(R)+120 °(S)

・B：240 °,300 °,360 °+0 °,-60 °(B′)
・B′：240 °,300 °,0 °,60 °+(-60 °)(B)
・H：180 °(I)+120 °,180 °(H)+(-60) °,-120 °(I′)
・I：180 °+120 °,180 °(H)+(-60) °,-120 °(I′)
・I′：180 °,240 °,300 °,360 °+240 °(I)

4. Coding technique

On a neighborhood in which jump is done, if coding is done using 2-fold symmetry, we can modify a code easily in case of occurring of a mistake. We assume seed point to be SP:2(the first convergence face J′K′L′M′), a neighborhood in which jump is done is chosen by two in a convergence face and they must be 2-fold symmetry.

/* inside side */
・(1) J′K′, L′M′, JK, LM
・(2) M′J′, K′L′, NJ, KL

/* inside vertex */
・(3) J′, L′, J, L
・(4) K′, M′, K, M

/* outside side */
・(5)AB, EF, A′B′, E′F′
・(6)BC, FG, B′C′, F′G′
・(7)CD, GH, C′D′, H′I′
・(8)DE, IA, D′E′, I′A′

/* outside vertex */
・(9) A, E, A′, E′
・(10) B, F, B′, F′
・(11) C, G, C′, G′
・(12) D, H, D′, I′

/* inside side */
・(13) O′P′, Q′R′, OP, QR
・(14) R′O′, P′Q′, SO, PQ

/* inside vertex */
・(15) O′, Q′, O, Q
・(16) P′, R′, P, R

5. Assignment
・Complete neighborhoods on sides and vertexes of Figure 2.

6. Modification of the past portion

We modify target pixel in ”Concrete example” in the last like following:

・target pixel：16

7. Concrete example

Figure 3 is a symmetrical graphics by Figure 2 and the following are data of program.

・SP:4(1st conv. face J′K′L′M′)+4(1st conv. face O′P′Q′R′)
・$n = 16$
・coordinates of painting number 0：member a:$(6(n-1)+1, 0(n-1)+1)$、member b:$(7(n-1)-1, 0(n-1)+1)$、member c:$(7(n-1)-1, 1(n-1)-1)$、member d:$(6(n-1)+1, 1(n-1)-1)$
・coordinates of painting number 0：member e:$(6(n-1)+1+3(n-1), 0(n-1)+1)$、member f:$(7(n-1)-1+3(n-1), 0(n-1)+1)$、member g:$(7(n-1)-1+3(n-1), 1(n-1)-1)$、member h:$(6(n-1)+1+3(n-1), 1(n-1)-1)$
・coordinates of painting number 1：member a:$\Delta x = 1$、member b:$\Delta y = 1$、member c:$\Delta x = -1$、member d:$\Delta y = -1$
・coordinates of painting number 1：member e:$\Delta x = 1$、member f:$\Delta y = 1$、member g:$\Delta x = -1$、member h:$\Delta y = -1$
If painting number 1 is finished at the first graphics, program pauses. Press Esc key.

・choice of CW, CCW：the same as the first
・painting algorithm：logical angle method
・painting timing：immediate painting
・push to stack：the same as the first

Array which is used for painting is initialized as follows:

・target pixel：16
・wall pixel：0

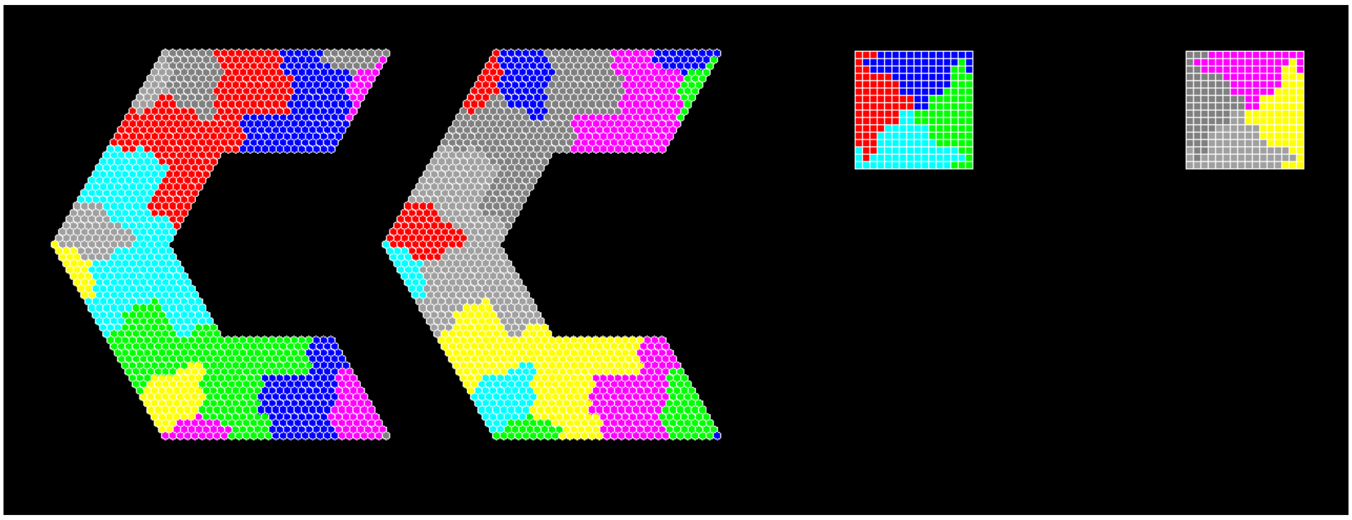


Figure 3

****

セルラーオートマトングラフィクス(9)

菊池盛雄

アブストラクト：
　正多面体を平面上に展開し、この展開図形に離散座標を設定し、正多面体の境界条件を適用して対称なグラフィクスを実現します。

1. ジョンソンの立体

　図１はジョンソンの立体のひとつである変形四角反柱を表しています。

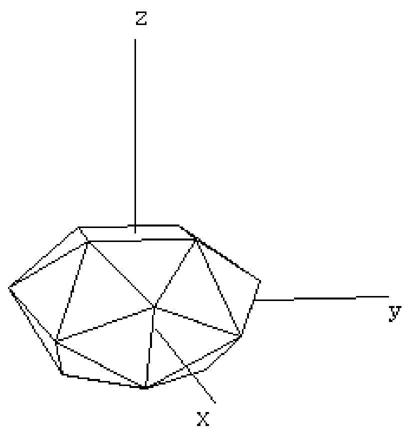


図１

この図形は多角形から成る鉢を向かい合わせでつなぎ合わせたような形をしています。この鉢は空間的に 45 °ずれています。この図形を平面に表したのが図２です。

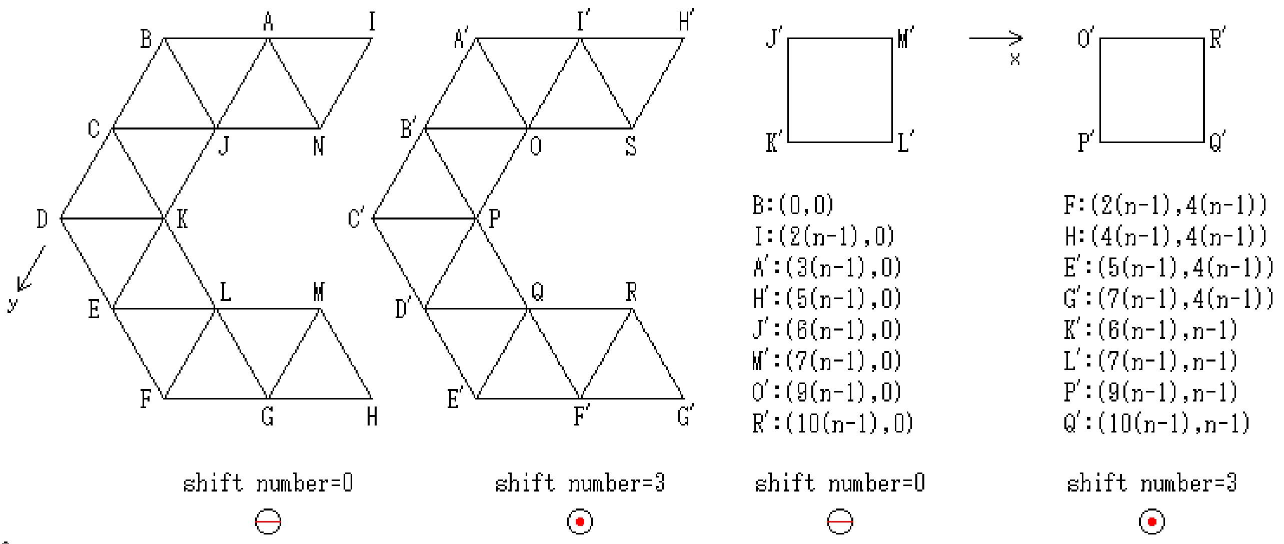


図２

上の鉢はシフト数が０であり、左のＣ型の図形が二次収束面、右の正方形が一次収束面です。下の鉢はシフト数が３であり、二次収束面、一次収束面は上の鉢と同じです。二次収束面は斜交座標、一次収束面は直交座標で表します。
　頂点の一致はダッシュを用いて表します。

・A-A′, B-B′, C-C′, D-D′, E-E′, F-F′, G-G′(H′), H(I)-I′
・J-J′, K-K′, L-L′, M(N)-M′, O-O′, P-P′, Q-Q′, R(S)-R′

二次収束面は一個所をカットして表しているので、M と N、H と I、R と S、G′ と H′ は同じ頂点です。

2. 対称性
　図 1 の図形の対称軸は z 軸です。シードポイントの配置の例を以下に示します。

・2 回対称：SP:2(上の一次収束面)、SP:2(上の一次収束面)+2(下の一次収束面)
・2 回対称：SP:2(上の二次収束面)、SP:2(上の二次収束面)+2(下の二次収束面)
・4 回対称：SP:4(上の一次収束面)、SP:4(上の一次収束面)+4(下の一次収束面)
・4 回対称：SP:4(上の二次収束面)、SP:4(上の二次収束面)+4(下の二次収束面)

上の一次収束面と下の一次収束面は空間的に 45 °ずれているので、SP:2+2 の対称グラフィクスは奇トンネルになりません。
　図 2 では便宜上図形を一次収束面と二次収束面に分解しました。実際にはこの二次収束面は二次収束面と三次収束面に分解することができます。

3. 近傍
　ジャンプが関与する近傍の例を角度で表します。

・MH：120 °,180 °,240 °,300 °+120 °,180 °(NI)
・NI：60 °,120 °,180 °,240 °+180 °,240 °(MH)
・RG′：120 °,180 °,240 °,300 °+120 °,180 °(SH′)
・SH′：60 °,120 °,180 °,240 °+180 °,240 °(RG′)

・JK：60 °,120 °,180 °,240 °+0 °(J′K′)
・J′K′：-90 °,0 °,90 °+120 °,180 °(JK)
・OP：60 °,120 °,180 °,240 °+0 °(O′P′)
・O′P′：-90 °,0 °,90 °+120 °,180 °(OP)

・AB：180 °,240 °,300 °,360 °+0 °,-60 °(A′B′)
・A′B′：-120 °,-60 °,0 °,60 °+(-60) °,-120 °(AB)

・J：0 °,60 °,120 °,180 °,240 °
・J′：-90 °,0 °+60 °,120 °,180 °(J)
・O：0 °,60 °,120 °,180 °,240 °
・O′：-90 °,0 °+60 °,120 °,180 °(O)
・M：180 °,240 °,300 °+120 °,180 °(N)
・N：60 °,120 °,180 °+180 °,240 °(M)
・M′：180 °,270 °+240 °,300 °(M)+120 °(N)
・R：180 °,240 °,300 °+120 °,180 °(S)
・S：60 °,120 °,180 °+180 °,240 °(R)
・R′：180 °,270 °+240 °,300 °(R)+120 °(S)

・B：240 °,300 °,360 °+0 °,-60 °(B′)

・B′：240 °,300 °,0 °,60 °+(-60 °)(B)
・H：180 °(I)+120 °,180 °(H)+(-60) °,-120 °(I′)
・I：180 °+120 °,180 °(H)+(-60) °,-120 °(I′)
・I′：180 °,240 °,300 °,360 °+240 °(I)

4. コーディング手法
　ジャンプが関与する近傍においては、2 回対称を用いてコーディングすればミスがあった場合コードの修正を容易に行うことができます。シードポイントは SP:2(一次収束面 J′K′L′M′) とし、ジャンプが関与する近傍は該当する収束面において二つ選び、いずれも 2 回対称とします。

/* 内側の辺 */
・(1) J′K′, L′M′, JK, LM
・(2) M′J′, K′L′, NJ, KL

/* 内側の点 */
・(3) J′, L′, J, L
・(4) K′, M′, K, M

/* 外側の辺 */
・(5)AB, EF, A′B′, E′F′
・(6)BC, FG, B′C′, F′G′
・(7)CD, GH, C′D′, H′I′
・(8)DE, IA, D′E′, I′A′

/* 外側の点 */
・(9) A, E, A′, E′
・(10) B, F, B′, F′
・(11) C, G, C′, G′
・(12) D, H, D′, I′

/* 内側の辺 */
・(13) O′P′, Q′R′, OP, QR
・(14) R′O′, P′Q′, SO, PQ

/* 内側の点 */
・(15) O′, Q′, O, Q
・(16) P′, R′, P, R

5. 課題
・図 2 の辺、頂点における近傍を完成させてください。

6. 過去分の修正
　前回の”具体例”におけるターゲットピクセルを以下のようにします。

・ターゲットピクセル：16

7. 具体例

　図 3 は図 2 による対称グラフィクスであり、以下はプログラムのデータです。

・SP:4(一次収束面 J′K′L′M′)+4(一次収束面 O′P′Q′R′)
・$n = 16$
・塗番号 0 の座標：メンバー a:$(6(n-1)+1, 0(n-1)+1)$、メンバー b:$(7(n-1)-1, 0(n-1)+1)$、メンバー c:$(7(n-1)-1, 1(n-1)-1)$、メンバー d:$(6(n-1)+1, 1(n-1)-1)$
・塗番号 0 の座標：メンバー e:$(6(n-1)+1+3(n-1), 0(n-1)+1)$、メンバー f:$(7(n-1)-1+3(n-1), 0(n-1)+1)$、メンバー g:$(7(n-1)-1+3(n-1), 1(n-1)-1)$、メンバー h:$(6(n-1)+1+3(n-1), 1(n-1)-1)$
・塗番号 1 の座標：メンバー a:$\Delta x = 1$、メンバー b:$\Delta y = 1$、メンバー c:$\Delta x = -1$、メンバー d:$\Delta y = -1$
・塗番号 1 の座標：メンバー e:$\Delta x = 1$、メンバー f:$\Delta y = 1$、メンバー g:$\Delta x = -1$、メンバー h:$\Delta y = -1$
最初のグラフィクスで塗番号 1 が終了するとプログラムが一時停止します。Esc キーを押してください。

・CW、CCW の選択：初回と同じ
・塗りつぶしアルゴリズム：論理角度法
・塗り方：即時塗りつぶし
・スタックへの座標のプッシュ：初回と同じ

塗りつぶしに用いられる配列は以下のように初期化します。

・ターゲットピクセル：16
・壁ピクセル：0

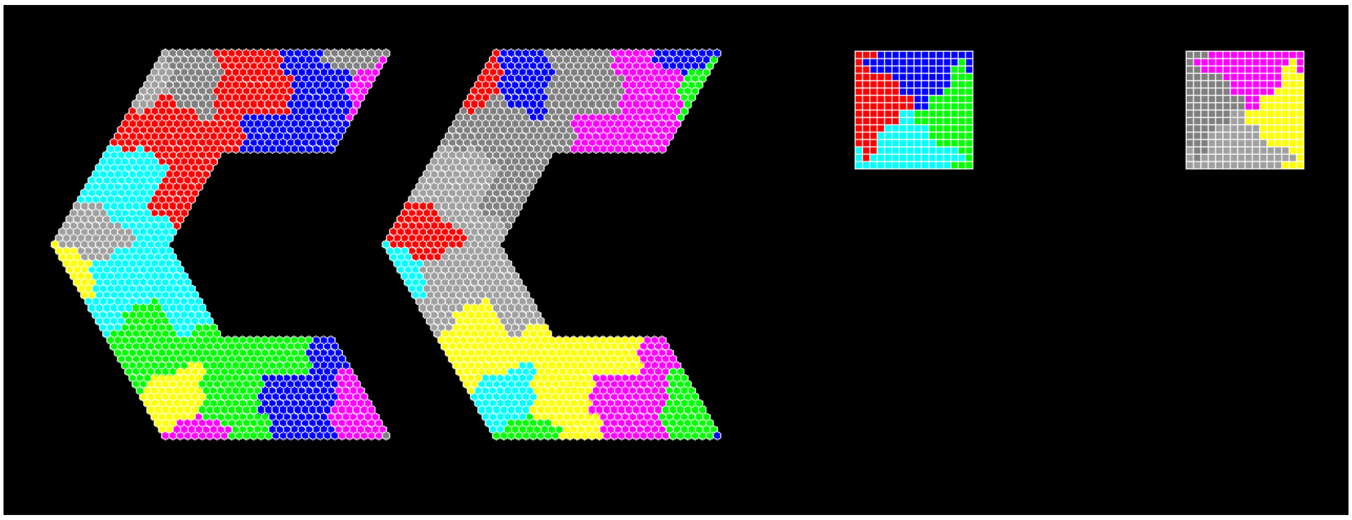


図 3

****

List 1:cag_9.c

```
/* t3.37                                                                    */
/* 2019 Morio Kikuchi                                                       */

#define WX /*0*//*1*/0              /* 0:Windows, 1:Xlib */

#if WX==0
#include <windows.h>
#else
#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include <X11/Xlocale.h>
#include <X11/cursorfont.h>
#include <X11/keysym.h>
#endif
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

#define VGACOLORS 50
#define dbl double
#define ASIZE_MS ((XRESO*YRESO)/CPMAX)  /* memory stack */
#define UdX 7
#define UdY 14
#if WX==0
#define POINT POINT
#define GKS GetKeyState
#define GKS_ GetKeyState
#define UDX (UdX+0)
#define UDY (UdY+1)
#else
#define FSIZE 14                    /* fontsize */
#define POINT XPoint
#define TRANS (65535./255)
#define XDSDY (asct+1)
#define UDX (UdX+0)
#define UDY (UdY+2)
#endif

#define CPMAX /*2*//*4*//*8*/8
#define RESO 16             /* n */
#define DSP (0)
```

```
#define YOUR_ART /*0*//*1*/0
#define RCMAX (394*2)

#if CPMAX==8
#define DIV 1
#elif CPMAX==4
#define DIV 0  /* CPMAX:4 => 0 */
#else
#define DIV 0
#endif

#if DIV==0
#define CPHALF CPMAX
#else
#define CPHALF (CPMAX/2)
#endif

#define XRESO 1280
#define YRESO 768
#define X0 (150)
#define Y0 (10)
#define dyMAX 603
#define GRPH_0_MAX 2000
#define SQSZ 24
#define PPDY (SQSZ+10)
#define DX_ 141

#define ICEIL(a,b) (((a)+((b)-1))/(b))
#define Zx (XRESO*2)
#define Zy (YRESO*2)
#define Zxd2 (Zx/2)
#define Zyd2 (Zy/2)

#define CD 38                       /* COLUMN_DIALOG */
#define ASIZE (256+1)
#define ASIZEM 256
#define DI 2
#define DJ 0
#define DI_d (DI+2)                 /* d : dialog */
#define DJ_d (DJ+2)
#define DI_m 1
#define dummy_R 'R'

char refill,pauseflag,fieldflag,GRPH,EDGE,c_trans,d_trans,clrpp,Fill,searchflag;
char charcode,charflag,zeroFill;
int Zflag,X,Y,X_,Y_,d0[2],xg,yg,zg,xi,yi,zi,id_1st;
int ca,c1,c2,c3,c4,c5,c7;
int x[8],y[8],x_[8],y_[8];
int enX[6+6],enY[6+6],enX_[6+6],enY_[6+6],enSN[6+6],jmp[6+6],id[Zxd2][Zyd2];
int ig,PIXSIZE,PIXSIZE_,idx,dy_hex,jmpflag,sn_,sn,bdrnum;
int xt,yt,ssize,tmp0,tmp1,putperiod;
long asize=ASIZE_MS;
long rcount[CPMAX],cnt;

char dialogflag,menuflag,filerflag,lumpflag_dialog,overwriteflag,
     BitBltflag_,noclearflag,bitbltflag,cut,BitBltflag,cqflag,insorover,overwriteflag,
     nocloseflag,passflag;
int icsr,jcsr,RTC=9,CSRDY=UDY;
long kmax_dialog,firstk_dialog;

char **pixel,p[ASIZE],p_dialog[ASIZE],p_restore[ASIZE],pixel_[XRESO][YRESO],
     cc[]="@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\\]^_";
long fp_mem[CPMAX],kmax;

char function,usflag;
unsigned char yorn;
int WB,WDX,WDY,Wdx,Wdy;
FILE *fp;

char dbflag,imm_restart_flag;
long dbsize;

#if WX==0
char immflag,imeendflag,compflag,dbcount;
#else
int argc_,asct;
char **argv_,appliname[]="CAG",fs1[ASIZE],stock_db[ASIZE];
long kmax_sdb;

char *fs2[5]={
              "-*-*-medium-r-normal--14*",       /* fn_set_0 (SFS) */
              "-*-*-medium-r-normal--20-*",       /* fn_set_1 (SFMM) */
              "-*-*-medium-r-normal--20-*",            /* fn_set_2 (SFM) */
              "-*-*-medium-r-normal--24-*",            /* fn_set_3 (SFL) */

              "none"
                                                };  /* fontnames */
#endif

typedef struct {
short CPM;long RCM;short xt1,xt2,yt;} coordinates;
coordinates xy;
```

```
typedef struct {
int cx,cy;} cag;
cag dsp[Zxd2][Zyd2];
cag bdr[XRESO];
typedef struct {
int x[CPMAX][10],y[CPMAX][10];} member;
member mbr[RCMAX];
typedef struct {
char p;dbl x,y,z;} swork;
swork work[Zx][Zy];
typedef struct {
int x,y;} sstack;
sstack stack[Zx*5]/*,stack_Rect[Zx*5]*/;
typedef struct {int xx,yy,xx_,yy_,sn;} ss;
ss s;ss rtn[CPMAX][ASIZE_MS];
typedef struct {
int back,fore;} bf;
bf bfset[]={{15,0},{0,15}};
#if WX==0
typedef struct {
unsigned char red,green,blue;} srgb;
srgb irgb[VGACOLORS];
#else
XColor irgb[VGACOLORS],c;
#endif

#if WX==0
HINSTANCE hinstance;
HWND hwnd;
HDC hdcdisplay,hdctmp1,hdctmp2,hdctmp3;
HBITMAP hbitmap1,hbitmap2,hbitmap3;
HPEN hpen;
HBRUSH hbrush;
HIMC himc,himc_;
COMPOSITIONFORM myime;
LOGFONT myimefont;
HFONT hfont;
POINT point;
RECT rect;
#else
Display *d;
int screen,depth;
Colormap cmap;
Window rw,w;
XSizeHints sh;
GC gcdisplay;
```

```
Pixmap pmap1,pmap2,pmap3;
XFontSet font_fs;
XFontStruct **info;
Cursor cursor;
XEvent event;
KeySym keysym,sym;
Visual *vis;
XImage *image;

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

void closegraph_(void),initpalette(void),BitBlt_full(void),setup(int),
     cleardevice_(char,int,int,int,int),rectangle_(char,int,int,int,int,int,int),
     delay_(long),beep(long),kbhit_(void),initgraph_return(void),
     use_subroop(void),keydowns_f2(void),bitblt(char,int,int,int,int,int,int),
     arrayreset(char,int),fwrite_mem(int),fread_mem(int),putpixel_(int,int,int),
     check_rcount(void),field(int),page_firstk_dialog(long),
     memcpy_(unsigned char*,long,unsigned char*,long,long),restore_in_PAINT(void);
char rpixel(int,int),csr_left_dialog(void),gettype_dialog(long);
unsigned char subroop(void);
int initgraph_(void),setup_(void),fourfloor_fiveceil(dbl),random_(int),
    getpixel_(int,int,int,int),cag_r(int,int,int),

    deletion_dialog(void),scroll_down_dialog(void),scroll_up_dialog(void),
    insertion_dialog(unsigned char),ishead_dialog(long);
long ftell_mem(int);
dbl getangle(int,int);

#if WX==0
COLORREF PALETTE(int);
LRESULT CALLBACK wndproc_by_kbhit_(HWND,UINT,WPARAM,LPARAM);
int wndproc_filer(HWND,UINT,WPARAM,LPARAM);
#else
int wndproc_filer(void);
XIMStyle InputStyle(XIM);
XIC InputContext(XIM,XIMStyle,XFontSet,Window);
#endif
```

```
int main(int argc,unsigned char **argv)
{
int k;
long mytime;
dbl val;

if(!YOUR_ART){
if(argc>1 && strcmp(argv[1],"0")==0){
GRPH=0;
if(argc==2) argc=1;else argc=2;
}
else GRPH=1;
}
else GRPH=1;
WB=1;
refill=1;
Zflag=1;
dO[0]=0;
dO[1]=0;

if(initgraph_()==1) return 1;

cleardevice_(1,0,0,XRESO,YRESO);
BitBlt_full();

xt=10*(RESO-1);yt=4*(RESO-1);
if(setup_()==1) return 1;
printf(" xt+1:%d\n",xt+1);
printf(" yt+1:%d\n",yt+1);
arrayreset(0,0);
arrayreset_();

if(argc>1) {time(&mytime);srand((unsigned int)mytime);}
else
srand(1);

while(1){
field(0);
/*putpixel(6*(RESO-1),0,8);*/
/*use_subroop();goto end;*/
cag_r(-1,-1,-1);
if(refill==0) break;
check_rcount();
/*printf(" %d\n",d_trans);*/

#if YOUR_ART==0
```

```
printf(" \n");
if(refill==0) break;
#else
delay_(2000);
c_trans=0;
arrayreset(1,-2);
field(-1);
/*BitBlt_full();*/
c_trans=3;
printf(" \n");
refill=2;
use_subroop();break;
#endif

if(GRPH){
beep(50);

delay_(6000);
if(pauseflag==1) {pauseflag=0;use_subroop();}
}/**if(GRPH)**/
if(refill==0) break;
arrayreset(2,0);
}/**while(1)**/

end:
closegraph_();

return 0;
}/** main **/



#if WX==0
void ls_image(char flag,char *file,int x,int y,int dx,int dy)
{
unsigned long xsize,ysize,size;
unsigned long width,height,imagesize;
unsigned long bits,bytesPerPixel,lineSizeDW,lineSize;
HDC hdce,hdc;
HBITMAP hbitmape;
BITMAPFILEHEADER bfh;
BITMAPINFOHEADER bih;
BYTE *gdata;
FILE *fpo,*fpi;

if(flag<=3){                        /* save */
if((fpo=fopen(file,"wb"))==NULL) {printf("Can't open a file.\n");return;}
```

```
width=dx;
height=dy;

bits=/*16*/24/*32*/;
bytesPerPixel=bits/8;
lineSizeDW=bytesPerPixel*width;
lineSizeDW=ICEIL(lineSizeDW,sizeof(long));
lineSize=lineSizeDW*sizeof(long);
imagesize=lineSize*height;

bfh.bfType=0x4d42;                 /* "BM" */
bfh.bfSize=54+imagesize;
bfh.bfReserved1=0;
bfh.bfOffBits=54;
bfh.bfReserved2=0;

bih.biSize=40;
bih.biWidth=width;
bih.biHeight=height;
bih.biPlanes=1;
bih.biBitCount=bits;
bih.biCompression=0;
bih.biSizeImage=imagesize;
bih.biXPelsPerMeter=0;
bih.biYPelsPerMeter=0;
bih.biClrUsed=0;
bih.biClrImportant=0;

if(flag<=1)
/*hdce=CreateCompatibleDC(hdctmp2)*/;
else if(flag==2)
hdce=CreateCompatibleDC(hdctmp1);
else{
hdc=CreateDC("DISPLAY",NULL,NULL,NULL);
hdce=CreateCompatibleDC(hdc);
}

hbitmape=CreateDIBSection(hdce,(LPBITMAPINFO)&bih,DIB_RGB_COLORS,&gdata,NULL,0);
SelectObject(hdce,hbitmape);

if(flag<=1)
/*BitBlt(hdce,0,0,dx,dy,hdctmp2,x,y,SRCCOPY)*/;
else if(flag==2)
BitBlt(hdce,0,0,dx,dy,hdctmp1,x,y,SRCCOPY);
else
```

```
BitBlt(hdce,0,0,dx,dy,hdc,x,y,SRCCOPY);

size=bih.biSizeImage;

fwrite(&bfh,14,1,fpo);
fwrite(&bih,40,1,fpo);
fwrite(gdata,size,1,fpo);

fclose(fpo);

if(flag==3) DeleteDC(hdc);
DeleteDC(hdce);
DeleteObject(hbitmape);

printf(" SAVE\n");
}
else{                                    /* load */
if((fpi=fopen(file,"rb"))==NULL) {printf("Can't open the file.\n");return;}

fread(&bfh,14,1,fpi);
if(bfh.bfType!=0x4d42) {fclose(fpi);printf("Not BM.\n");return;}
fread(&bih,40,1,fpi);

fseek(fpi,bfh.bfOffBits,0);
size=bih.biSizeImage;
gdata=(BYTE *)malloc(size);
fread(gdata,size,1,fpi);

/*StretchDIBits(hdctmp2,x,y,bih.biWidth,bih.biHeight,0,0,bih.biWidth,bih.biHeight,
              gdata,(LPBITMAPINFO)&bih,DIB_RGB_COLORS,SRCCOPY);*/

fclose(fpi);
free(gdata);
}
}/** ls_image **/
#else
void ls_image(char flag,char *file,int x,int y,int dx,int dy)
{
unsigned long xsize,ysize,size;
unsigned long unitbytes,width,height,bits_per_pixel,bytes_per_line;
unsigned long i,j,k,k_,knew,oddbytes,dksum;
int c0,c1,c2;
unsigned long long heightdiv2,dbx;
unsigned char *buf_,*buf,*bf,*swap;
FILE *fpo,*fpi;
```

```
typedef struct {
unsigned char bfType[2];
unsigned long bfSize;
unsigned short bfReserved1;
unsigned short bfOffBits;
unsigned long bfReserved2;
} bfhset;
bfhset bfh;

typedef struct {
unsigned long biSize;
unsigned long biWidth;
unsigned long biHeight;
unsigned short biPlanes;
unsigned short biBitCount;
unsigned long biCompression;
unsigned long biSizeImage;
unsigned long biXPelsPerMeter;
unsigned long biYPelsPerMeter;
unsigned long biClrUsed;
unsigned long biClrImportant;
} bihset;
bihset bih;

if(flag<=3){                        /* save */
if(depth==16)      {unitbytes=2;printf(" 16bpp\n");}
else if(depth==24) {unitbytes=4;printf(" 24bpp\n");}
else               {printf("Depth unsuitable.\n");return;}

if((fpo=fopen(file,"wb"))==NULL) {printf("Can't open a file.\n");return;}

if(flag<=1)
/*image=XGetImage(d,pmap2,x,y,dx,dy,AllPlanes,ZPixmap)*/;
else if(flag==2)
image=XGetImage(d,pmap1,x,y,dx,dy,AllPlanes,ZPixmap);
else
image=XGetImage(d,rw,x,y,dx,dy,AllPlanes,ZPixmap);

width=image->width;
height=image->height;

oddbytes=(XRESO*3)%4;
if(oddbytes==0){
buf=(unsigned char *)malloc(1L*XRESO*3*YRESO);
swap=(unsigned char *)malloc(XRESO*3);
}
```

```
else{
buf=(unsigned char *)malloc(1L*(XRESO*3+(4-oddbytes))*YRESO);
swap=(unsigned char *)malloc(XRESO*3+(4-oddbytes));
}

if((oddbytes=(width*3)%4)==0){
size=width*height;
for(i=0;i<size;i++){
if(unitbytes==4){                    /* 4:24bpp, 2:16bpp */
buf[i*3+0]=image->data[i*4+0];
buf[i*3+1]=image->data[i*4+1];
buf[i*3+2]=image->data[i*4+2];
}
else{
c0=image->data[i*2+1];
c1=image->data[i*2+0];
                                     /* red, green, blue */
buf[i*3+2]=((c0 >> 3) & 31)*255/31;
buf[i*3+1]=(((c0 & 0x07) << 2) | ((c1 >> 6) & 0x03))*255/31;
buf[i*3+0]=(c1 & 31)*255/31;
}
}

bytes_per_line=width*3;
}/**if(oddbytes)**/
else{
k=0;k_=0;dksum=0;
for(j=0;j<height;j++){
for(i=0;i<width;i++){
if(unitbytes==4){
buf[k*3+0+dksum]=image->data[k_*4+0];
buf[k*3+1+dksum]=image->data[k_*4+1];
buf[k*3+2+dksum]=image->data[k_*4+2];
}
else{
c0=image->data[k_*2+1];
c1=image->data[k_*2+0];
                                     /* red, green, blue */
buf[k*3+2+dksum]=((c0 >> 3) & 31)*255/31;
buf[k*3+1+dksum]=(((c0 & 0x07) << 2) | ((c1 >> 6) & 0x03))*255/31;
buf[k*3+0+dksum]=(c1 & 31)*255/31;
}

k++;k_++;
}
if(unitbytes==2) k_+=(width*3)%2;   /* 16bpp */

knew=k*3+0+dksum;
for(i=0;i<4-oddbytes;i++){
buf[knew]=0;

knew++;
}

dksum+=(4-oddbytes);
}/**for(j)**/

bytes_per_line=width*3+(4-oddbytes);
}/**else(oddbytes)*/

/*printf(" size=%ld\n",bytes_per_line*height);
printf(" %d %d %d\n",width,height,width*height*3);*/

/*99*/
strcpy(bfh.bfType,"BM");
/*bfh.bfSize=bytes_per_line*height/65536;*/
bfh.bfSize=54+bytes_per_line*height;
bfh.bfReserved1=0;
bfh.bfOffBits=54;
bfh.bfReserved2=0;

bih.biSize=40;
bih.biWidth=width;
bih.biHeight=height;
bih.biPlanes=1;
bih.biBitCount=8*3;                 /* 24bpp */
bih.biCompression=0;
bih.biSizeImage=bytes_per_line*height;
bih.biXPelsPerMeter=2925;
bih.biYPelsPerMeter=2925;
bih.biClrUsed=0;
bih.biClrImportant=0;

size=bih.biSizeImage;
heightdiv2=height/2;
dbx=bytes_per_line;
for(i=0;i<heightdiv2;i++){
memmove(swap,&buf[i*dbx],dbx);
memmove(&buf[i*dbx],&buf[size-(i+1)*dbx],dbx);
memmove(&buf[size-(i+1)*dbx],swap,dbx);
}
```

```
fwrite(&bfh,14,1,fpo);
fwrite(&bih,40,1,fpo);
size=bih.biSizeImage;
fwrite(buf,size,1,fpo);

fclose(fpo);
free(buf);
free(swap);

printf(" SAVE\n");
}
else{                              /* load */
if(depth==16)      {printf("16bpp\n");}
else if(depth==24) {printf("24bpp\n");}
else               {printf("Depth unsuitable.\n");return;}

if((fpi=fopen(file,"rb"))==NULL) {printf("Can't open the file.\n");return;}

fread(&bfh,14,1,fpi);
if(strncmp(bfh.bfType,"BM",2)!=0) {fclose(fpi);printf("Not BM.\n");return;}
fread(&bih,40,1,fpi);

fseek(fpi,bfh.bfOffBits,0);
size=bih.biSizeImage;
buf_=(unsigned char *)malloc(size);
fread(buf_,size,1,fpi);

fclose(fpi);

width=bih.biWidth;
height=bih.biHeight;
bits_per_pixel=bih.biBitCount;
bytes_per_line=bih.biSizeImage/bih.biHeight;

oddbytes=(width*3)%4;
if(oddbytes==0)
swap=(unsigned char *)malloc(width*3);
else
swap=(unsigned char *)malloc(width*3+(4-oddbytes));

size=bih.biSizeImage;
heightdiv2=height/2;
dbx=bytes_per_line;
for(i=0;i<heightdiv2;i++){
memmove(swap,&buf_[i*dbx],dbx);
memmove(&buf_[i*dbx],&buf_[size-(i+1)*dbx],dbx);
memmove(&buf_[size-(i+1)*dbx],swap,dbx);
}

buf=(unsigned char *)malloc(width*height*3);
bf=(unsigned char *)malloc(width*height*4);

if((oddbytes=(width*3)%4)==0){
size=width*height;
for(i=0;i<size;i++){
buf[i*3+0]=buf_[i*3+0];
buf[i*3+1]=buf_[i*3+1];
buf[i*3+2]=buf_[i*3+2];
}
}/**if(oddbytes)**/
else{
k=0;k_=0;dksum=0;
for(j=0;j<height;j++){
for(i=0;i<width;i++){
buf[k*3+0]=buf_[k_*3+0+dksum];
buf[k*3+1]=buf_[k_*3+1+dksum];
buf[k*3+2]=buf_[k_*3+2+dksum];

k++;k_++;
}

dksum+=(4-oddbytes);
}/**for(j)**/
}/**if(oddbytes)**/

if(depth==16){
size=width*height;
for(i=0;i<size;i++){
/*c0=buf[i*3+0]*31/255;
c1=buf[i*3+1]*31/255;
c2=buf[i*3+2]*31/255;

buf[i*3+0]=0;
buf[i*3+1]=(c0<<3) | ((c1>>2) & 0x07);
buf[i*3+2]=(((c1 & 0x03) << 6) | c2)+32;*/

c0=buf[i*3+2]*31/255;                   /* bmp */
c1=buf[i*3+1]*31/255;
c2=buf[i*3+0]*31/255;

buf[i*3+2]=0;
buf[i*3+1]=(c0<<3) | ((c1>>2) & 0x07);
buf[i*3+0]=(((c1 & 0x03) << 6) | c2)+32;
}
}/**if(depth)**/

size=width*height;
for(i=0;i<size;i++){
bf[i*4+0]=buf[i*3+0];
bf[i*4+1]=buf[i*3+1];
bf[i*4+2]=buf[i*3+2];
}

vis=DefaultVisual(d,screen);

image=XCreateImage(d,vis,depth,ZPixmap,0,bf,width,height,32,width*4);

image->byte_order=LSBFirst;
image->bitmap_bit_order=LSBFirst;
image->bits_per_pixel=8*4;

/*XPutImage(d,pmap2,gcdisplay,image,0,0,x,y,width,height);*/

free(buf_);
free(buf);
free(bf);
free(swap);
}
}/** ls_image **/
#endif


void fprintf_(char *str,int v2,int v3,int v4,int v5,int v6)
{
FILE *fp;

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

fprintf(fp," %s %d %d %d %d %d\n",str,v2,v3,v4,v5,v6);

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


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;
}/** use_subroop **/


unsigned char subroop(void)
{
charflag=1;

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


#if WX==0
void keydowns_f2(void)
{
int dy;

if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0) charflag=0;
else if(GKS('S')<0){
dy=Y0+(yt+1/*1.7*/)*PIXSIZE+15+PPDY;
ls_image(2,"ss.bmp",0,0,XRESO,/*YRESO*/dy);
beep(300);
}
}/** keydowns_f2 **/
#else
void keydowns_f2(void)
{
int dy;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0) charflag=0;
else if(GKS('S')<0 || GKS('s')<0){
dy=Y0+(yt+1/*1.7*/)*PIXSIZE+15+PPDY;
ls_image(2,"ss.bmp",0,0,XRESO,/*YRESO*/dy);
beep(300);
```

```
}
}/** keydowns_f2 **/
#endif


void restore_in_PAINT(void)
{
#if WX==0
ValidateRect(hwnd,NULL);
#endif

bitblt(1,0,0,XRESO,YRESO,0,0);
if(dialogflag) {BitBlt_dialog(2);csr();}
}/** restore_in_PAINT **/


#if WX==1
long max(long a,long b)
{
return (a>b)?a:b;
}/** max **/


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


void setup(int flag)
{
if(flag==0){
if(YOUR_ART==0) EDGE=1;
else            EDGE=0;
Fill=-1;
}
#if WX==0
else if(flag==1){
WDX=GetSystemMetrics(SM_CXFIXEDFRAME);
WDY=GetSystemMetrics(SM_CYCAPTION)+GetSystemMetrics(SM_CYFIXEDFRAME);
}
else{
GetWindowRect(hwnd,&rect);
Wdx=rect.left;
Wdy=rect.top;
```

```
}
#endif
}/** setup **/


int get_dx_h(int nx,int ny)
{
int dx;

dx=ff_fc(X0+nx*1.0*PIXSIZE_-ny*0.5*PIXSIZE_);

/*if(nx<7*(RESO-1)) return (dx-200);
else              return (dx-300);*/
}/** get_dx_h **/


int get_dy_h(int nx,int ny)
{
int dy;

dy=ff_fc(Y0+ny*(sqrt(3)/2)*PIXSIZE_);

return dy;
}/** get_dy_h **/


int setup_(void)
{
int i,dy,n,m;

PIXSIZE=10;
dy=Y0+(yt+1/*1.7*/)*PIXSIZE+10;

if(dy>dyMAX){
while(1){
PIXSIZE--;
dy=Y0+(yt+1/*1.7*/)*PIXSIZE+10;
if(dy<=dyMAX) break;
}
}

if(PIXSIZE<4) PIXSIZE=4;
if(1) PIXSIZE=7;
PIXSIZE_=/*ff_fc(PIXSIZE*sqrt(2))*/PIXSIZE;

dy=get_dy_h(1,1)-get_dy_h(0,0);
```

```
dy_hex=ff_fc(dy/3.);

pixel=(/*unsigned */char **)malloc(sizeof(/*unsigned */char *)*((xt+1)+1));
if(pixel==NULL){
initgraph_return();return 1;}

i=0;
while(1){
pixel[i]=(/*unsigned */char *)malloc(sizeof(/*unsigned */char)*((yt+1)+1));

if(pixel[i]==NULL){
while(1){
i--;
if(i<0) break;
free(pixel[i]);
}
free(pixel);
initgraph_return();return 1;}

i++;
if(i==(xt+1)+1) break;
}

return 0;
}/** setup_ **/


#if WX==0
void initsysfont(int type)
{
if(type==0){
hfont=CreateFont(UdY,UdX,0,0,
                 FW_NORMAL,FALSE,0,0,
                 DEFAULT_CHARSET,OUT_DEFAULT_PRECIS,
                 CLIP_DEFAULT_PRECIS,DEFAULT_QUALITY,
                 FIXED_PITCH | FF_ROMAN/*FF_MODERN*/,NULL);
}
else{
hfont=CreateFont(UdY,UdX,0,0,
                 FW_NORMAL,0,0,0,
                 DEFAULT_CHARSET,OUT_DEFAULT_PRECIS,
                 CLIP_DEFAULT_PRECIS,DEFAULT_QUALITY,
                 FIXED_PITCH | /*FF_ROMAN*/FF_MODERN,NULL);
}

SelectObject(hdcdisplay,hfont);
SelectObject(hdctmp1,hfont);
SelectObject(hdctmp2,hfont);
}/** initsysfont **/
#else
void initsysfont(int type)
{
if(type==0){                         /* small */
strcpy(fs1,fs2[0]);
strcat(fs1,"-*-*-*-*-*-*");          /* scalable */

font_fs=XCreateFontSet(d,fs1,&mlist,&mcount,&def);
XFontsOfFontSet(font_fs,&info,&flist);
asct=(*info)->ascent;
}
else if(type==-1){                   /* medium (math) */
strcpy(fs1,fs2[1]);
strcat(fs1,"-*-*-*-*-*-*");          /* scalable */

font_fs=XCreateFontSet(d,fs1,&mlist,&mcount,&def);
XFontsOfFontSet(font_fs,&info,&flist);
asct=(*info)->ascent;
}
else if(type==1){                    /* medium */
strcpy(fs1,fs2[2]);
strcat(fs1,"-*-*-*-*-*-*");          /* scalable */

font_fs=XCreateFontSet(d,fs1,&mlist,&mcount,&def);
XFontsOfFontSet(font_fs,&info,&flist);
asct=(*info)->ascent;
}
else{                                /* large */
strcpy(fs1,fs2[3]);
strcat(fs1,"-*-*-*-*-*-*");          /* scalable */

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

if(mcount>0)
font_fs=XCreateFontSet(d,"-*-*-medium-r-normal--14-*",&mlist,&mcount,&def);

XFontsOfFontSet(font_fs,&info,&flist);
asct=(*info)->ascent;
}
}/** initsysfont **/
#endif
```

```
int initgraph_(void)
{
#if WX==0
WNDCLASS wndclass;

setup(0);
hinstance=GetModuleHandle(NULL);

wndclass.hInstance    =hinstance;
wndclass.lpszClassName="CAGCLASS";
wndclass.lpszMenuName =NULL;
wndclass.lpfnWndProc  =wndproc_by_kbhit_;
wndclass.style        =0;
wndclass.hIcon        =LoadIcon(hinstance,"MYICON");
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) return 1;

hwnd=CreateWindow("CAGCLASS"," CAG",
                  /*WS_POPUP,*/
                  WS_OVERLAPPED | WS_CAPTION | WS_SYSMENU | WS_MINIMIZEBOX,
                  0,0,XRESO,YRESO,
                  NULL,NULL,hinstance,NULL);
if(hwnd==NULL) {MessageBox(NULL,"Memory space is not left.","CAG",MB_OK);return 1;}

SetWindowPos(hwnd,HWND_TOP,0,0,0,0,SWP_NOMOVE | SWP_NOSIZE);
ShowWindow(hwnd,SW_SHOWDEFAULT);

hdcdisplay=GetDC(hwnd);

hbitmap1=CreateCompatibleBitmap(hdcdisplay,XRESO,YRESO);
hbitmap2=CreateCompatibleBitmap(hdcdisplay,XRESO,YRESO);
hbitmap3=CreateCompatibleBitmap(hdcdisplay,XRESO,UDY);

hdctmp1=CreateCompatibleDC(hdcdisplay);  /* text, dialog, menu */
hdctmp2=CreateCompatibleDC(hdcdisplay);  /* text, dialog, menu */
hdctmp3=CreateCompatibleDC(hdcdisplay);  /* cursor */

SelectObject(hdctmp1,hbitmap1);
SelectObject(hdctmp2,hbitmap2);
SelectObject(hdctmp3,hbitmap3);

SetBkMode(hdcdisplay,TRANSPARENT);
SetBkMode(hdctmp1,TRANSPARENT);
SetBkMode(hdctmp2,TRANSPARENT);
SetBkMode(hdctmp3,TRANSPARENT);

SetBkColor(hdcdisplay,PALETTE(bfset[WB].back));
SetBkColor(hdctmp1,PALETTE(bfset[WB].back));
SetBkColor(hdctmp2,PALETTE(bfset[WB].back));
SetBkColor(hdctmp3,PALETTE(bfset[WB].back));
#else
if((d=XOpenDisplay(""))==NULL) return 1;
screen=DefaultScreen(d);
cmap=DefaultColormap(d,screen);

setup(0);

rw=DefaultRootWindow(d);
w=XCreateSimpleWindow(d,rw,0,0,XRESO,YRESO,0,
                      irgb[bfset[WB].back].pixel,
                      irgb[bfset[WB].back].pixel);
sh.flags=PPosition | PSize;
sh.x=0;sh.y=0;
sh.width=XRESO;sh.height=YRESO;
XSetStandardProperties(d,w,appliname,appliname,None,argv_,argc_,&sh);

XSelectInput(d,w,KeyPressMask | ButtonPressMask | PointerMotionMask | ExposureMask);
XMoveWindow(d,w,0,0);
XMapWindow(d,w);
XFlush(d);

gcdisplay=XCreateGC(d,w,0,NULL);

depth=DefaultDepth(d,screen);
pmap1=XCreatePixmap(d,w,XRESO,YRESO,depth);  /* text, dialog, menu */
pmap2=XCreatePixmap(d,w,XRESO,YRESO,depth);  /* text, dialog, menu */
pmap3=XCreatePixmap(d,w,XRESO,UDY,depth);  /* cursor */

cursor=XCreateFontCursor(d,XC_arrow);
XDefineCursor(d,w,cursor);

XSetLineAttributes(d,gcdisplay,/*2*/1,LineSolid,CapButt,JoinMiter);
#endif

initsysfont(0);
```

```
#if WX==1
initIME();
#endif
initpalette();
setstccolor(bfset[WB].fore);
setcsrcolor(15);

setup(1);

return 0;
}/** initgraph_ **/


#if WX==0
void initgraph_return(void)
{
DeleteObject(hfont);
DeleteDC(hdctmp1);
DeleteDC(hdctmp2);
DeleteDC(hdctmp3);
DeleteObject(hbitmap1);
DeleteObject(hbitmap2);
DeleteObject(hbitmap3);

/*EndPaint(hwnd,&paintstruct);*/
ReleaseDC(hwnd,hdcdisplay);
DestroyWindow(hwnd);
/*UnregisterClass("CAGCLASS",hinstance);*/

MessageBox(NULL,"Memory space is not left.","CAG",MB_OK);
}/** initgraph_return **/
#else
void initgraph_return(void)
{
XFreeFontSet(d,font_fs);
XFreeCursor(d,cursor);
XFreePixmap(d,pmap1);
XFreePixmap(d,pmap2);
XFreePixmap(d,pmap3);
XFreeGC(d,gcdisplay);

XFreeColormap(d,cmap);
XDestroyWindow(d,w);XFlush(d);
XCloseDisplay(d);

printf(" %s\n","Memory not enough");
```

```
}/** initgraph_return **/
#endif


void closegraph_(void)
{
int i;

i=0;
while(1){
free(pixel[i]);
i++;
if(i==(xt+1)+1) break;
}
free(pixel);

#if WX==0
DeleteObject(hfont);
DeleteObject(hbitmap1);
DeleteObject(hbitmap2);
DeleteObject(hbitmap3);
DeleteDC(hdctmp1);
DeleteDC(hdctmp2);
DeleteDC(hdctmp3);

/*EndPaint(hwnd,&paintstruct);*/
ReleaseDC(hwnd,hdcdisplay);
DestroyWindow(hwnd);
/*UnregisterClass("CAGCLASS",hinstance);*/
#else
XFreeFontSet(d,font_fs);
XFreeCursor(d,cursor);
XFreePixmap(d,pmap1);
XFreePixmap(d,pmap2);
XFreePixmap(d,pmap3);
XFreeGC(d,gcdisplay);

XFreeColormap(d,cmap);
XDestroyWindow(d,w);XFlush(d);
XCloseDisplay(d);
#endif
}/** closegraph_ **/


void initpalette(void)
{
```

```
int i;

irgb[0].red=0;irgb[0].green=0;irgb[0].blue=0;

irgb[9].red=0;irgb[9].green=0;irgb[9].blue=255;  /* blue */
irgb[10].red=0;irgb[10].green=255;irgb[10].blue=0;  /* green */
irgb[11].red=0;irgb[11].green=255;irgb[11].blue=255;  /* cyan */
irgb[12].red=255;irgb[12].green=0;irgb[12].blue=0;  /* red */
irgb[13].red=255;irgb[13].green=0;irgb[13].blue=255;  /* magenta */
irgb[14].red=255;irgb[14].green=255;irgb[14].blue=0;  /* yellow */

irgb[15].red=255;irgb[15].green=255;irgb[15].blue=255;

for(i=7;i<9;i++){                  /* 7, 8 */
irgb[i].red=128+32*(8-i);
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}

for(i=16;i<20;i++){                /* 16 -> 19 */
irgb[i].red=255-24*(20-i);
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}

for(i=1;i<7;i++){                  /* 1 -> 6 */
if(irgb[9+(i-1)].red==255)
irgb[i].red=irgb[9+(i-1)].red-24*1;
if(irgb[9+(i-1)].green==255)
irgb[i].green=irgb[9+(i-1)].green-24*1;
if(irgb[9+(i-1)].blue==255)
irgb[i].blue=irgb[9+(i-1)].blue-24*1;
}

for(i=20;i<26;i++){                /* 20 -> 25 */
if(irgb[9+(i-20)].red==255)
irgb[i].red=irgb[9+(i-20)].red-24*2;
if(irgb[9+(i-20)].green==255)
irgb[i].green=irgb[9+(i-20)].green-24*2;
if(irgb[9+(i-20)].blue==255)
irgb[i].blue=irgb[9+(i-20)].blue-24*2;
}

for(i=26;i<32;i++){                /* 26 -> 31 */
if(irgb[9+(i-26)].red==255)
irgb[i].red=irgb[9+(i-26)].red-24*3;
if(irgb[9+(i-26)].green==255)
irgb[i].green=irgb[9+(i-26)].green-24*3;
if(irgb[9+(i-26)].blue==255)
irgb[i].blue=irgb[9+(i-26)].blue-24*3;
}

for(i=32;i<38;i++){                 /* 32 -> 37 */
if(irgb[9+(i-32)].red==255)
irgb[i].red=irgb[9+(i-32)].red-24*4;
if(irgb[9+(i-32)].green==255)
irgb[i].green=irgb[9+(i-32)].green-24*4;
if(irgb[9+(i-32)].blue==255)
irgb[i].blue=irgb[9+(i-32)].blue-24*4;
}

for(i=38;i<44;i++){                 /* 38 -> 43 */
if(irgb[9+(i-38)].red==255)
irgb[i].red=irgb[9+(i-38)].red-24*5;
if(irgb[9+(i-38)].green==255)
irgb[i].green=irgb[9+(i-38)].green-24*5;
if(irgb[9+(i-38)].blue==255)
irgb[i].blue=irgb[9+(i-38)].blue-24*5;
}

for(i=44;i<50;i++){                 /* 44 -> 49 */
if(irgb[9+(i-44)].red==255)
irgb[i].red=irgb[9+(i-44)].red-24*6;
if(irgb[9+(i-44)].green==255)
irgb[i].green=irgb[9+(i-44)].green-24*6;
if(irgb[9+(i-44)].blue==255)
irgb[i].blue=irgb[9+(i-44)].blue-24*6;
}

#if WX==1
for(i=0;i<VGACOLORS;i++){
irgb[i].red=fourfloor_fiveceil(irgb[i].red*TRANS);
irgb[i].green=fourfloor_fiveceil(irgb[i].green*TRANS);
irgb[i].blue=fourfloor_fiveceil(irgb[i].blue*TRANS);
}

for(i=0;i<VGACOLORS;i++)
XAllocColor(d,cmap,&irgb[i]);

XParseColor(d,cmap,"cyan",&c);
XAllocColor(d,cmap,&c);
#endif
```

```
}/** initpalette **/


void BitBlt_full(void)
{
bitblt(1,0,0,XRESO,YRESO,0,0);
}/** BitBlt_full **/


#if WX==0
void bitblt(char flag,int x,int y,int xsize,int ysize,int x_,int y_)
{
int dy_=0;

if(bitbltflag==0){
if(flag==1)
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp1,x,y,SRCCOPY);
else if(flag==2)
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp2,x,y,SRCCOPY);
else if(flag==3)                    /* flag = 3 */ /* for csr() */
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp2,/*x*/x_,/*y*/y_-dy_,SRCCOPY);
}/**if(bitbltflag)**/
else{
if(flag==1)
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp1,x,y,SRCINVERT);
else if(flag==2)
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp2,x,y,SRCINVERT);
else if(flag==-3)              /* flag = -3 */ /* for csr_to_1(), for csr_to_1_BL() */
BitBlt(hdctmp2,x_,y_-dy_,xsize,ysize,
       hdctmp3,x,y,SRCINVERT);         /* x, y = 0, 0 */
}/**else(bitbltflag)**/
}/** bitblt **/
#else
void bitblt(char flag,int x,int y,int xsize,int ysize,int x_,int y_)
{
if(bitbltflag==0){
if(flag==1){
/*printf(" %d %d %d %d %d %d\n",x,y,x_,y_,xsize,ysize);*/
XCopyArea(d,pmap1,w,gcdisplay,x,y,xsize,ysize,
                              x_,y_);
}
else if(flag==2)
XCopyArea(d,pmap2,w,gcdisplay,x,y,xsize,ysize,
                                  x_,y_);
else if(flag==3)                    /* flag = 3 */ /* for csr() */
XCopyArea(d,pmap2,w,gcdisplay,/*x*/x_,/*y*/y_,xsize,ysize,
                              x_,y_);
}/**if(bitbltflag)**/
else{
if(flag==1)
XCopyArea(d,pmap1,w,gcdisplay,x,y,xsize,ysize,
                              x_,y_);
else if(flag==2)
XCopyArea(d,pmap2,w,gcdisplay,x,y,xsize,ysize,
                                  x_,y_);
else if(flag==-3)              /* flag = -3 */ /* for csr_to_1(), for csr_to_1_BL() */
XCopyArea(d,pmap3,pmap2,gcdisplay,/*x*/0,/*y*/0,xsize,ysize,
                                  x_,y_);
}/**else(bitbltflag)**/

XFlush(d);
}/** bitblt **/
#endif



#if WX==0
void cleardevice_(char hdc,int x,int y,int xsize,int ysize)
{
if(hdc==0)
PatBlt(hdcdisplay,x,y,xsize,ysize,PALETTE(bfset[WB].back));
else if(hdc==1)
PatBlt(hdctmp1,x,y,xsize,ysize,PALETTE(bfset[WB].back));
else
PatBlt(hdctmp2,x,y,xsize,ysize,PALETTE(bfset[WB].back));
}/** cleardevice_ **/
#else
void cleardevice_(char hdc,int x,int y,int xsize,int ysize)
{
XSetForeground(d,gcdisplay,irgb[bfset[WB].back].pixel);

if(hdc==0)
XFillRectangle(d,w,gcdisplay,x,y,xsize,ysize);
else if(hdc==1)
XFillRectangle(d,pmap1,gcdisplay,x,y,xsize,ysize);
else
XFillRectangle(d,pmap2,gcdisplay,x,y,xsize,ysize);
}/** cleardevice_ **/
```

```
#endif


#if WX==0
void setstccolor(int color)
{
SetTextColor(hdcdisplay,PALETTE(color));
SetTextColor(hdctmp1,PALETTE(color));
SetTextColor(hdctmp2,PALETTE(color));
}/** setstccolor **/
#else
void setstccolor(int color)
{
XSetForeground(d,gcdisplay,irgb[color].pixel);
}/** setstccolor **/
#endif


#if WX==0
void stc(char hdc,int x,int y,unsigned char *str,int size)
{
if(hdc!=0)
TextOut(hdcdisplay,x,y,str,size);

if(hdc==0)
TextOut(hdcdisplay,x,y,str,size);
else if(hdc==1)
TextOut(hdctmp1,x,y,str,size);
else
TextOut(hdctmp2,x,y,str,size);
}/** stc **/
#else
void stc(char hdc,int x,int y,unsigned char *str,int size)
{
if(hdc!=0)
XmbDrawString(d,w,font_fs,gcdisplay,x,y+XDSDY,str,size);

if(hdc==0)
XmbDrawString(d,w,font_fs,gcdisplay,x,y+XDSDY,str,size);
else if(hdc==1)
XmbDrawString(d,pmap1,font_fs,gcdisplay,x,y+XDSDY,str,size);
else
XmbDrawString(d,pmap2,font_fs,gcdisplay,x,y+XDSDY,str,size);
}/** stc **/
#endif
```

```
#if WX==0
char gettype(char flag,unsigned char s1,long k,long kend)
{
char type;

/* code page:932(SJIS)-> */
if(s1>=0x20 && s1<=0x7e) type=0;                     /* word */
else if(s1>=0xa1 && s1<=0xdf) 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;                           /* others */
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 */

return type;
}/** gettype **/
#else
char gettype(char flag,unsigned char s1,long k,long kend)
{
char type;

/* EUC-> */
if(s1>=0x20 && s1<=0x7e) type=0;                     /* word */
else if(flag==1 && k==kend) type=0;
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<=0x7f) type=-1;                           /* others */
else type=3;                                         /* Double Byte Character */
/* <-EUC */
/*else if(s1>=0x81 && s1<=0xfc && (s1<=0x9f || s1>=0xe0) &&
        s2>=0x40 && s2<=0xfc && s2!=0x7f) type=3;*/  /* Double Byte Character */

return type;
}/** gettype **/
#endif

int while_puts_show_(int xlast,int ylast)
{
char type;
int i,j,dx,dy;
```

```
long k;
unsigned char s[1];
unsigned char jis[2];

i=xlast;j=ylast;
k=0;

while(1){
s[0]=p[k];
type=gettype(1,s[0],k,kmax);

if(type<=2){
dx=(i+0)*UDX;dy=(j+0)*UDY;
stc(1,dx,dy,s,1);

if(k==kmax) break;

k++;

i++;
}/**if(type)**/
else if(type==3){
jis[0]=p[k];
jis[1]=p[k+1];

dx=(i+0)*UDX;dy=(j+0)*UDY;
stc(1,dx,dy,jis,2);

if(/*k==kmax-1 || */k==kmax) break;              /* ? */

k+=2;
}/**else if(type)**/
else{
}/**else(type)**/

}

return i;
}/** while_puts_show_ **/


void printf_(char *str,dbl val,long i,long j)
{
char pflag=1;
int dx,dy;
int columns,len1,len2;
```

```
unsigned char buf[11];

dx=i*UDX;dy=j*UDY;

if(pflag==1) columns=7;
else         columns=11;

/*itoa(val,buf,10);*/gcvt(val,columns-1,buf);
len1=strlen(str);
len2=strlen(buf);

cleardevice_(0,dx,dy,UDX*(len1+7),UDY);  /* instead of bitblt(pflag,...) */
if(pflag==1)
paint(pflag,dx,dy,UDX*(len1+7),UDY+0,bfset[WB].back);

/* string */
setstccolor(bfset[WB].fore);
/*stc(1,dx,dy,str,strlen(str));*/
strcpy(p,str);kmax=strlen(p)-1;
while_puts_show_(i,j);

/* value */
stc(pflag,dx+len1*UDX+2,dy,buf,len2);

if(/*pflag==1*/0)
bitblt(pflag,dx,dy,UDX*(len1+7),UDY,dx,dy);
}/** printf_ **/


#if WX==0
void paint(char hdc,int x,int y,int xsize,int ysize,int color)
{
hbrush=CreateSolidBrush(PALETTE(color));

if(hdc==0){
SelectObject(hdcdisplay,hbrush);
PatBlt(hdcdisplay,x,y,xsize,ysize,PATCOPY);
}
else if(hdc==1){
SelectObject(hdctmp1,hbrush);
PatBlt(hdctmp1,x,y,xsize,ysize,PATCOPY);
}
else{
SelectObject(hdctmp2,hbrush);
PatBlt(hdctmp2,x,y,xsize,ysize,PATCOPY);
}
```

```
DeleteObject(hbrush);
}/** paint **/
#else
void paint(char hdc,int x,int y,int xsize,int ysize,int color)
{
XSetForeground(d,gcdisplay,irgb[color].pixel);

if(hdc==0)
XFillRectangle(d,w,gcdisplay,x,y,xsize,ysize);
else if(hdc==1)
XFillRectangle(d,pmap1,gcdisplay,x,y,xsize,ysize);
else
XFillRectangle(d,pmap2,gcdisplay,x,y,xsize,ysize);
}/** paint **/
#endif



#if WX==0
COLORREF PALETTE(int color)
{
return RGB(irgb[color].red,irgb[color].green,irgb[color].blue);
}/** PALETTE **/
#endif



#if WX==0
void kbhit_(void)
{
MSG msg;

if(PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
TranslateMessage(&msg);
DispatchMessage(&msg);
}
}/** kbhit_ */
#else
int GKS(KeySym XK)
{
if(keysym==XK) return -1;
else return 0;
}/** GKS **/


int GKS_(long ModkeyMask)
{
```

```
if((event.xkey.state & ModkeyMask)>0) return -1;
else return 0;
}/** GKS_ **/


void kbhit_(void)
{
if(XPending(d)){
XNextEvent(d,&event);

wndproc_filer();
}
}/** kbhit_ */
#endif


#if WX==0
LRESULT CALLBACK  wndproc_by_kbhit_(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
if(wndproc_filer(hwnd,umsg,wparam,lparam)!=0) return 1;

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


#if WX==0
int wndproc_filer(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
char BDflag,old,old_,gotoflag;
int x[2],y[2],z[2],dlt=SQSZ,val,dx,dy,i,j;
static int pcolor_old=8,pcolor=9,procedure=0;
POINT cpos;

setup(2);

if(umsg==WM_KEYDOWN){
/************************* 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(2);csr();}
else if(BitBltflag_==1)             csr();
else{}

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

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

if(function==2){
imm_pause();
keydowns_f2();
return 1;
}

if(usflag==1) usflag=0;

     if(GKS(VK_ESCAPE)<0 || GKS(VK_PAUSE)<0) refill=0;
```

```
else if(GKS(VK_SHIFT)<0) pauseflag=1;

return 1;
}/**else if(umsg)**/
else if(umsg==WM_SYSKEYDOWN){
}/**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 if(umsg==WM_LBUTTONDOWN){
if(YOUR_ART==0) return 1;
if(refill==1 || dialogflag>0) return 1;

GetCursorPos(&cpos);
```

```
dx=cpos.x-WDX-Wdx;dy=cpos.y-WDY-Wdy;
x[0]=dsp[dx][dy].cx;y[0]=dsp[dx][dy].cy;

if(x[0]<0 || y[0]<0){
if(Fill==-1){
if(procedure==0){
BDflag=getflag(dx,dy);
if(BDflag==3) return 1;
}
else{
rectangle_(0,19*dlt,dlt-PPDY+2,(19+2)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
procedure=0;
return 1;
}
}/**if(Fill)**/
else{
if(procedure==0){
BDflag=getflag(dx,dy);
if(BDflag==3) return 1;
}
else{
rectangle_(0,26*dlt,dlt-PPDY+2,(26+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
procedure=0;
return 1;
}
}/**else(Fill)**/
}/**if(x[0],y[0])**/

if(Fill==-1){
if(procedure==0){                     /* one action */
BDflag=getflag(dx,dy);

if(BDflag==0 && xg==17 && clrpp==1){
rectangle_(0,17*dlt,dlt-PPDY+2,(17+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
}

if(BDflag==0){
if(xg==16){
restore_edge();
rectangle_(0,16*dlt,dlt-PPDY+2,(16+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
printf(" Restoration of edge\n");
}
else if(xg==17){
if(!clrpp) clrpp=1;else clrpp=0;
printf_("clrpp=",clrpp,DX_,2);
printf(" color PlusPlus\n");
}
else{
pcolor_old=pcolor;
pcolor=xg;
printf(" color=%d old=%d\n",xg,pcolor_old);
}
}/**if(BDflag)**/
else if(BDflag==1){
     if(xg==0) procedure=1;            /* ID_1st */
else if(xg==1) procedure=2;            /* Clear */
else if(xg==2){
field(-1);
rectangle_(0,21*dlt,dlt-PPDY+2,(21+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
printf(" All Clear\n");
}
else if(xg==3){
refill=1;
fsave("");
rectangle_(0,22*dlt,dlt-PPDY+2,(22+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
refill=2;
}
else if(xg==4){
refill=1;
fload("");
rectangle_(0,23*dlt,dlt-PPDY+2,(23+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
refill=2;
}

     if(xg==0)
printf(" change ID_1st\n");
else if(xg==1)
printf(" Clear\n");
}/**else if(BDflag)**/
else if(BDflag==2){
     if(xg==0){
Fill=0;bdrnum=0;
printf(" into Fill(Fill:%d)\n",Fill);
}
}/**else if(BDflag)**/
else{
search_i(x[0],y[0],pcolor);
}/**else(BDflag)**/
}/**if(procedure)**/
else if(procedure==1){
if(x[0]>=0 && y[0]>=0 && (val=pixel[x[0]][y[0]])>=0 && val<=15){
id_1st=id[x[0]][y[0]];              /* change 1st */
```

```
/*BitBlt_full();*/
printf_("id_1st=",id_1st,DX_,0);
printf(" new ID_1st=%d\n",id_1st);
}
else{                                /* id[][]=-2, -1 */
printf(" unpointed\n");
}

rectangle_(0,19*dlt,dlt-PPDY+2,(19+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
procedure=0;
}/**else if(procedure)**/
else if(procedure==2){
/*if(x[0]<0 || y[0]<0) printf(" ?\n");*/
if(x[0]>=0 && y[0]>=0 && (val=pixel[x[0]][y[0]])>=0 && val<=15){
id[x[0]][y[0]]=-1;                   /* restore one */
old=EDGE;EDGE=1;
search_i(x[0],y[0],16);
EDGE=old;
/*BitBlt_full();*/
}
else{
printf(" unpointed\n");
}

rectangle_(0,20*dlt,dlt-PPDY+2,(20+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
procedure=0;
}/**else if(procedure)**/
}/**if(Fill)*************************************************************/
else{
if(procedure==0){
BDflag=getflag(dx,dy);

if(BDflag==0 && xg==17 && clrpp==1){
rectangle_(0,17*dlt,dlt-PPDY+2,(17+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
}

if(BDflag==0){
if(xg==16){
restore_edge();
rectangle_(0,16*dlt,dlt-PPDY+2,(16+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
printf(" Restoration of edge\n");
}
else if(xg==17){
if(!clrpp) clrpp=1;else clrpp=0;
printf_("clrpp=",clrpp,DX_,2);
printf(" color PlusPlus\n");
```

```
}
else{
pcolor_old=pcolor;
pcolor=xg;
printf(" color=%d old=%d\n",xg,pcolor_old);
}
}/**if(BDflag)**/
else if(BDflag==1){
}/**else if(BDflag)**/
else if(BDflag==2){
     if(xg==0){
Fill=-1;
C_and_S(-1,-1,-1);
printf(" out of Fill(Fill:%d)\n",Fill);
}
else if(xg==1) procedure=1;         /* C */
else if(xg==2){                     /* AC */
Fill=0;
C_and_S(-1,-1,-1);
bdrnum=0;
rectangle_(0,(25+2)*dlt,dlt-PPDY+2,(25+3)*dlt,dlt-PPDY+3,bfset[WB].back,0);/* erase */
}
else if(xg==3){                     /* R */
/*refill=1;*/
fload("tmp.bin");
rectangle_(0,28*dlt,dlt-PPDY+2,(28+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
/*refill=2;*/
}
}/**else if(BDflag)**/
else{
if(/*GKS_(VK_CONTROL)<0*/1){
if(Fill==0){
search_bdr(x[0],y[0]);
}
else{
if((i=bdrcheck(x[0],y[0]))<0) search_bdr(x[0],y[0]);
}
}/**if(GKS_(VK_CONTROL))**/
}/**else(BDflag)**/
}/**if(procedure)**/
else if(procedure==1){
for(i=0;i<bdrnum;i++){
if(bdr[i].cx==x[0] && bdr[i].cy==y[0]){
old_=Fill;Fill=0;
val=pixel[bdr[i].cx][bdr[i].cy];    /* restore one(magenta and white) */
if(val!=16)
```

```
search_i(bdr[i].cx,bdr[i].cy,val);
else{
old=EDGE;EDGE=1;
search_i(bdr[i].cx,bdr[i].cy,val);
EDGE=old;
}
Fill=old_;

for(j=i+1;j<bdrnum;j++){
bdr[j-1].cx=bdr[j].cx;
bdr[j-1].cy=bdr[j].cy;
}
bdrnum--;

/*BitBlt_full();*/
break;
}
}/**for(i)**/

if(i==bdrnum) printf(" unpointed\n");

rectangle_(0,(25+1)*dlt,dlt-PPDY+2,(25+2)*dlt,dlt-PPDY+3,bfset[WB].back,0);/* erase */
procedure=0;
}/**else if(procedure)**/
}/**else(Fill)***************************************************************/

return 1;
}/**else if(umsg)**/
else if(umsg==WM_RBUTTONDOWN){  /* R_BUTTON */
if(YOUR_ART==0) return 1;
if(refill==1 || dialogflag>0) return 1;
if(Fill<0) return 1;

GetCursorPos(&cpos);
dx=cpos.x-WDX-Wdx;dy=cpos.y-WDY-Wdy;
x[0]=dsp[dx][dy].cx;y[0]=dsp[dx][dy].cy;

if(x[0]<0 || y[0]<0) return 1;

     if(Fill>0 && GKS_(VK_CONTROL)<0 && GKS_(VK_SHIFT)>=0) val=0;  /* C+0 */
else if(Fill>0 && GKS_(VK_CONTROL)>=0 && GKS_(VK_SHIFT)<0) val=1;  /* 0+S */
else if(Fill>0 && GKS_(VK_CONTROL)>=0 && GKS_(VK_SHIFT)>=0) val=2;  /* C+S */
else if(Fill>0 && GKS_(VK_CONTROL)<0 && GKS_(VK_SHIFT)<0) val=3;  /* C+S */
else if(Fill==0) val=4;  /* 0+S */
Fill=0;
xg=x[0];yg=y[0];
```

```
     if(val==3) {val=2;C_and_S(val,pcolor,pcolor_old);}
else if(val==4) {val=3;C_and_S(val,pcolor,-1);}
else                   C_and_S(val,pcolor,pcolor);
bdrnum=0;
printf(" done\n");

return 1;
}/**else if(umsg)**/
else if(umsg==WM_MOUSEMOVE){
if(YOUR_ART==0) return 1;
if(refill==1 || dialogflag>0 || Fill==-1) return 1;
if(!(wparam & MK_LBUTTON)) return 1;

GetCursorPos(&cpos);
dx=cpos.x-WDX-Wdx;dy=cpos.y-WDY-Wdy;
x[0]=dsp[dx][dy].cx;y[0]=dsp[dx][dy].cy;

if(x[0]<0 || y[0]<0) return 1;

if(/*GKS_(VK_CONTROL)<0*/1){
if(Fill==0){
search_bdr(x[0],y[0]);
}
else{
if((i=bdrcheck(x[0],y[0]))<0) search_bdr(x[0],y[0]);
}
}/**if(GKS_(VK_CONTROL))**/

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

return 0;
}/** wndproc_filer **/
#else
int wndproc_filer(void)
{
char gotoflag,buf[10],BDflag,old,old_;
int length,x[2],y[2],z[2],dlt=SQSZ,val,dx,dy,i,j;
static int pcolor_old=8,pcolor=9,procedure=0;
unsigned char buf_Xmb[ASIZE];
Window root,child;
int rx,ry,wx,wy;
unsigned int mask;

entrance:
/*length=*/XLookupString((XKeyEvent *)&event,buf,10,&keysym,NULL);
```

```
/*buf[length]='\0';*/

length=XmbLookupString(ic,(XKeyEvent *)&event,buf_Xmb,ASIZE,&sym,&st);
if(st==XLookupBoth || st==XLookupChars){}
else length=0;
if(length>0) buf_Xmb[length]='\0';

if(dialogflag>0) imm_restart();

if(event.type==KeyPress){
/************************* menu keydowns -> ***************************/
/************************* <- menu keydowns ***************************/

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

if(dialogflag>0){

if(usflag==1) usflag=0;

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

gotoflag=1;

if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0 || GKS(XK_F12)<0 || GKS(XK_F1)<0){
  imm_pause();
  dialogflag=3;refill=0;BitBltflag_=2;}
else if(GKS(XK_Return)<0){
  imm_pause();
  trim_dialog();
  dialogflag=2;refill=0;BitBltflag_=2;}

else {gotoflag=0;}

if(gotoflag==1) goto end_dialog;

end_left_dialog:
left_keydowns_dialog();

end_dialog:
if(length>0){
if(length>1)
```

```
WM_funcIME_CHAR(buf_Xmb);                  /* double byte */
else if(length==1 && GKS_(ControlMask)>=0 && GKS_(Mod1Mask)>=0 &&
        buf_Xmb[0]>=0x20 && buf_Xmb[0]<=0x7e)
WM_func_CHAR(buf_Xmb[0]);           /* only ASCII CODE */
else
WM_func_CHAR(0);
}

if(BitBltflag_==0) {BitBlt_dialog(2);csr();}
else if(BitBltflag_==1)             csr();
else{}

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

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

if(function==2){
keydowns_f2();
return 1;
}

if(usflag==1) usflag=0;

     if(GKS(XK_Escape)<0 || GKS(XK_Pause)<0) refill=0;
else if(GKS(XK_Shift_L)<0 || GKS(XK_Shift_R)<0) pauseflag=1;

return 1;
}/**if(event.type)**/
else if(event.type==Expose){
restore_in_PAINT();

return 1;
}/**else if(event.type)**/
else if(event.type==ButtonPress){
if(YOUR_ART==0) return 1;
if(refill==1 || dialogflag>0) return 1;

dx=event.xbutton.x;dy=event.xbutton.y;
x[0]=dsp[dx][dy].cx;y[0]=dsp[dx][dy].cy;

if(event.xbutton.button==1){        /* L_BUTTON */
if(x[0]<0 || y[0]<0){
if(Fill==-1){
if(procedure==0){
BDflag=getflag(dx,dy);
if(BDflag==3) return 1;
}
else{
rectangle_(0,19*dlt,dlt-PPDY+2,(19+2)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
procedure=0;
return 1;
}
}/**if(Fill)**/
else{
if(procedure==0){
BDflag=getflag(dx,dy);
if(BDflag==3) return 1;
}
else{
rectangle_(0,26*dlt,dlt-PPDY+2,(26+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
procedure=0;
return 1;
}
}/**else(Fill)**/
}/**if(x[0],y[0])**/

if(Fill==-1){
if(procedure==0){
BDflag=getflag(dx,dy);

if(BDflag==0 && xg==17 && clrpp==1){
rectangle_(0,17*dlt,dlt-PPDY+2,(17+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
}

if(BDflag==0){
if(xg==16){
restore_edge();
rectangle_(0,16*dlt,dlt-PPDY+2,(16+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
printf(" Restoration of edge\n");
}
else if(xg==17){
if(!clrpp) clrpp=1;else clrpp=0;
printf_("clrpp=",clrpp,DX_,2);
printf(" color PlusPlus\n");
}
else{
pcolor_old=pcolor;
pcolor=xg;
printf(" color=%d old=%d\n",xg,pcolor_old);
}
}/**if(BDflag)**/
```

```
else if(BDflag==1){
     if(xg==0) procedure=1;            /* ID_1st */
else if(xg==1) procedure=2;            /* Clear */
else if(xg==2){
field(-1);
rectangle_(0,21*dlt,dlt-PPDY+2,(21+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
printf(" All Clear\n");
}
else if(xg==3){
refill=1;
fsave("");
rectangle_(0,22*dlt,dlt-PPDY+2,(22+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
refill=2;
}
else if(xg==4){
refill=1;
fload("");
rectangle_(0,23*dlt,dlt-PPDY+2,(23+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
refill=2;
}

     if(xg==0)
printf(" change ID_1st\n");
else if(xg==1)
printf(" Clear\n");
}/**else if(BDflag)**/
else if(BDflag==2){
     if(xg==0){
Fill=0;bdrnum=0;
printf(" into Fill(Fill:%d)\n",Fill);
}
}/**else if(BDflag)**/
else{
search_i(x[0],y[0],pcolor);
}/**else(BDflag)**/
}/**if(procedure)**/
else if(procedure==1){
if(x[0]>=0 && y[0]>=0 && (val=pixel[x[0]][y[0]])>=0 && val<=15){
id_1st=id[x[0]][y[0]];              /* change 1st */
/*BitBlt_full();*/
printf_("id_1st=",id_1st,DX_,0);
printf(" new ID_1st=%d\n",id_1st);
}
else{                               /* id[][]=-2, -1 */
printf(" unpointed\n");
}
```

```
rectangle_(0,19*dlt,dlt-PPDY+2,(19+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
procedure=0;
}/**else if(procedure)**/
else if(procedure==2){
/*if(x[0]<0 || y[0]<0) printf(" ?\n");*/
if(x[0]>=0 && y[0]>=0 && (val=pixel[x[0]][y[0]])>=0 && val<=15){
id[x[0]][y[0]]=-1;                  /* restore one */
old=EDGE;EDGE=1;
search_i(x[0],y[0],16);
EDGE=old;
/*BitBlt_full();*/
}
else{
printf(" unpointed\n");
}

rectangle_(0,20*dlt,dlt-PPDY+2,(20+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
procedure=0;
}/**else if(procedure)**/
}/**if(Fill)*****************************************************************/
else{
if(procedure==0){
BDflag=getflag(dx,dy);

if(BDflag==0 && xg==17 && clrpp==1){
rectangle_(0,17*dlt,dlt-PPDY+2,(17+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
}

if(BDflag==0){
if(xg==16){
restore_edge();
rectangle_(0,16*dlt,dlt-PPDY+2,(16+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
printf(" Restoration of edge\n");
}
else if(xg==17){
if(!clrpp) clrpp=1;else clrpp=0;
printf_("clrpp=",clrpp,DX_,2);
printf(" color PlusPlus\n");
}
else{
pcolor_old=pcolor;
pcolor=xg;
printf(" color=%d old=%d\n",xg,pcolor_old);
}
}/**if(BDflag)**/
```

```
else if(BDflag==1){
}/**else if(BDflag)**/
else if(BDflag==2){
     if(xg==0){
Fill=-1;
C_and_S(-1,-1,-1);
printf(" out of Fill(Fill:%d)\n",Fill);
}
else if(xg==1) procedure=1;         /* C */
else if(xg==2){                     /* AC */
Fill=0;
C_and_S(-1,-1,-1);
bdrnum=0;
rectangle_(0,(25+2)*dlt,dlt-PPDY+2,(25+3)*dlt,dlt-PPDY+3,bfset[WB].back,0);/* erase */
}
else if(xg==3){                     /* R */
/*refill=1;*/
fload("tmp.bin");
rectangle_(0,28*dlt,dlt-PPDY+2,(28+1)*dlt,dlt-PPDY+3,bfset[WB].back,0);  /* erase */
/*refill=2;*/
}
}/**else if(BDflag)**/
else{
if(/*GKS_(ControlMask)<0*/1){
if(Fill==0){
search_bdr(x[0],y[0]);
}
else{
if((i=bdrcheck(x[0],y[0]))<0) search_bdr(x[0],y[0]);
}
}/**if(GKS_(ControlMask))**/
}/**else(BDflag)**/
}/**if(procedure)**/
else if(procedure==1){
for(i=0;i<bdrnum;i++){
if(bdr[i].cx==x[0] && bdr[i].cy==y[0]){
old_=Fill;Fill=0;
val=pixel[bdr[i].cx][bdr[i].cy];    /* restore one(magenta and white) */
if(val!=16)
search_i(bdr[i].cx,bdr[i].cy,val);
else{
old=EDGE;EDGE=1;
search_i(bdr[i].cx,bdr[i].cy,val);
EDGE=old;
}
Fill=old_;
for(j=i+1;j<bdrnum;j++){
bdr[j-1].cx=bdr[j].cx;
bdr[j-1].cy=bdr[j].cy;
}
bdrnum--;

/*BitBlt_full();*/
break;
}
}/**for(i)**/

if(i==bdrnum) printf(" unpointed\n");

rectangle_(0,(25+1)*dlt,dlt-PPDY+2,(25+2)*dlt,dlt-PPDY+3,bfset[WB].back,0);/* erase */
procedure=0;
}/**else if(procedure)**/
}/**else(Fill)***************************************************************/
}/**if(event.xbutton.button)**/
else{                               /* R_BUTTON */
if(Fill<0) return 1;
if(x[0]<0 || y[0]<0) return 1;

     if(Fill>0 && GKS_(ControlMask)<0 && GKS_(ShiftMask)>=0) val=0;  /* C+0 */
else if(Fill>0 && GKS_(ControlMask)>=0 && GKS_(ShiftMask)<0) val=1;  /* 0+S */
else if(Fill>0 && GKS_(ControlMask)>=0 && GKS_(ShiftMask)>=0) val=2;  /* C+S */
else if(Fill>0 && GKS_(ControlMask)<0 && GKS_(ShiftMask)<0) val=3;  /* C+S */
else if(Fill==0) val=4;  /* 0+S */
Fill=0;
xg=x[0];yg=y[0];
     if(val==3) {val=2;C_and_S(val,pcolor,pcolor_old);}
else if(val==4) {val=3;C_and_S(val,pcolor,-1);}
else                  C_and_S(val,pcolor,pcolor);
bdrnum=0;
printf(" done\n");
}/**else(event.xbutton.button)**/

return 1;
}/**else if(event.type)**/
else if(event.type==MotionNotify){
if(YOUR_ART==0) return 1;
if(refill==1 || dialogflag>0 || Fill==-1) return 1;
XQueryPointer(d,w,&root,&child,&rx,&ry,&wx,&wy,&mask);
if(!(mask & Button1Mask)) return 1;

dx=event.xbutton.x;dy=event.xbutton.y;
```

```
x[0]=dsp[dx][dy].cx;y[0]=dsp[dx][dy].cy;

if(x[0]<0 || y[0]<0) return 1;

if(/*GKS_(ControlMask)<0*/1){
if(Fill==0){
search_bdr(x[0],y[0]);
}
else{
if((i=bdrcheck(x[0],y[0]))<0) search_bdr(x[0],y[0]);
}
}/**if(GKS_(ControlMask))**/

return 1;
}/**else if(event.type)**/

return 0;
}/** wndproc_filer **/
#endif


void write_bdr(int i,char color)
{
pixel_[bdr[i].cx][bdr[i].cy]=color;
}/** write_bdr **/


void C_and_S(char flag,int color,int color_old)
{
int i,pcolor,old,old_,val=2;
static int nest=0;

nest++;

if(flag==-1){                       /* restore all(magenta and white) */
old_=Fill;Fill=0;

for(i=0;i<bdrnum;i++){
val=pixel[bdr[i].cx][bdr[i].cy];
if(val==16){                        /* 16 first */
old=EDGE;EDGE=1;
search_i(bdr[i].cx,bdr[i].cy,16);
EDGE=old;
}
}/**for(i)**/
```

```
zi=1;
for(i=0;i<bdrnum;i++){
val=pixel[bdr[i].cx][bdr[i].cy];
if(val!=16){
search_i(bdr[i].cx,bdr[i].cy,val);
}
}/**for(i)**/
zi=0;

Fill=old_;
}
else if(flag==0){                    /* C+0 */
if(nest==1) fsave("tmp.bin");
old=c_trans;c_trans=2;searchflag=1;

for(i=0;i<bdrnum;i++)
search_i(bdr[i].cx,bdr[i].cy,color);

c_trans=old;searchflag=0;
}
else if(flag==1){                    /* 0+S */
fsave("tmp.bin");
old=c_trans;c_trans=1;

field(0);
for(i=0;i<bdrnum;i++) write_bdr(i,0);
if(pixel_[xg][yg]==16){
val=cag_r(xg,yg,color);              /* 0:good, 1:bad */
if(val==0){
field(0);
c_trans=2;
for(i=0;i<bdrnum;i++) write_bdr(i,0);
cag_r(xg,yg,color);
}

refill=2;
}/**if(pixel_[][])**/

c_trans=old;

C_and_S(-1,-1,-1);                   /* (-1,;restore all(magenta and white) */
}
else if(flag==2){                    /* C+S */
fsave("tmp.bin");
old=c_trans;c_trans=1;
```

```
field(0);
for(i=0;i<bdrnum;i++) write_bdr(i,0);
if(pixel_[xg][yg]==16){             /* 16 > 15 */
val=cag_r(xg,yg,color);             /* 0:good, 1:bad */
if(val==0){
field(0);
c_trans=2;
for(i=0;i<bdrnum;i++) write_bdr(i,0);
cag_r(xg,yg,color);
}

refill=2;
}/**if(pixel_[][])**/

c_trans=old;

if(val==0)
C_and_S(0,color_old,-1);
else
C_and_S(-1,-1,-1);                  /* (-1,:restore all(magenta and white) */
}
else if(flag==3){                   /* no border(magenta and white) */
fsave("tmp.bin");
old=c_trans;c_trans=1;

if((zg=pixel[xg][yg])!=color){
c_trans=2;
zeroFill=1;
cag_r(xg,yg,color);
zeroFill=0;

refill=2;
}/**if(pixel[][])**/

c_trans=old;

C_and_S(-1,-1,-1);                  /* (-1,;restore all(magenta and white) */
}

nest=0;
printf_("id_1st=",id_1st,DX_,0);
}/** C_and_S **/


int bdrcheck(int x,int y)
{
```

```
int i;

for(i=0;i<bdrnum;i++){
if(x==bdr[i].cx && y==bdr[i].cy) return i;
}

return -1;
}/** bdrcheck **/


void delay_(long millisecond)
{
long oldtime,nowtime,dtime,old;
dbl i=CLOCKS_PER_SEC,j;

j=millisecond;
millisecond=j*(i/1000.);
oldtime=clock();

while(1){
kbhit_();
if(pauseflag==1 && refill==0) {pauseflag=0;refill=1;break;}
if(refill==0) break;

old=dtime;
nowtime=clock();dtime=nowtime-oldtime;
if(dtime>=millisecond) break;
if(dtime<0){
millisecond-=old;
oldtime=0;
}
}
}/** delay_ **/


void beep(long millisecond)
{
#if WX==0
Beep(888,millisecond);
#endif
}/** beep **/


int fourfloor_fiveceil(dbl val_d)
{
int val_i,val;
```

```
val_i=floor(val_d);
val=(val_d-val_i<0.5)?val_i:val_i+1;

return val;
}/** fourfloor_fiveceil **/


int ff_fc(dbl val_d)
{
return fourfloor_fiveceil(val_d);
}/** ff_fc **/


void arrayreset(char flag,int wcolor)
{
int i,j,k;

if(flag<=1){                        /* 0, 1 */
i=0;
while(1){

j=0;
while(1){
pixel[i][j]=wcolor;
j++;
if(j==(yt+1)+1) break;
}

i++;
if(i==(xt+1)+1) break;
}
}/**if(flag)**/

if(flag==0){                        /* 0 */
for(i=0;i<Zxd2;i++)
for(j=0;j<Zyd2;j++){
dsp[i][j].cx=-2;dsp[i][j].cy=-2;
}
}/**if(flag)**/

if(flag!=1){                        /* 0, 2 */
for(i=0;i<RCMAX;i++)  /* rcount[CPMAX-1] */
for(j=0;j<CPMAX;j++)  /* ig */
for(k=0;k<10;k++){    /* putpixel() + pp_() */
mbr[i].x[j][k]=-1;
```

```
mbr[i].y[j][k]=-1;
}

for(i=0;i<Zxd2;i++)
for(j=0;j<Zyd2;j++)
id[i][j]=-2;
}/**if(flag)**/
}/** arrayreset **/


void arrayreset_(void)
{
int i,j;

i=0;
while(1){

j=0;
while(1){
pixel_[i][j]=0;
j++;
if(j==(yt+1)+1) break;
}

i++;
if(i==(xt+1)+1) break;
}
}/** arrayreset_ **/


void restore_work(char flag)
{
int i;

if(flag==0){
if(idx){
for(i=idx-1;i>-1;i--){
work[stack[i].x][stack[i].y].p=0;
}

idx=0;
}
}
else{
/*if(idx_Rect){
for(i=idx_Rect-1;i>-1;i--){
```

```
work_Rect[stack_Rect[i].x][stack_Rect[i].y]=0;
}

idx_Rect=0;
}*/
}
}/** restore_work **/


#if WX==0
int ppixel(int nx,int ny,int pcolor)
{
if((nx<0)||(nx>XRESO-1)||(ny<0-PPDY)||(ny>YRESO-1)) return 1;
/*if((nx<0)||(nx>XRESO-1)||(ny<0)||(ny>YrESO-1)) return 1;*/

ny+=PPDY;

SetPixelV(hdcdisplay,nx,ny,PALETTE(pcolor));
if(1){
SetPixelV(hdctmp1,nx,ny,PALETTE(pcolor));
if(fieldflag==0 && c_trans==0){
dsp[nx][ny].cx=xi;dsp[nx][ny].cy=yi;  /* nx, ny:display coordinates */
}
}
else{
SetPixelV(hdctmp1,nx,ny,PALETTE(pcolor));
if(xg==0){
SetPixelV(hdctmp1,nx+1,ny,PALETTE(pcolor));
SetPixelV(hdctmp1,nx-1,ny,PALETTE(pcolor));
}
else{
SetPixelV(hdctmp1,nx,ny+1,PALETTE(pcolor));
SetPixelV(hdctmp1,nx,ny-1,PALETTE(pcolor));
}
}

return 0;
}/** ppixel **/
#else
int ppixel(int nx,int ny,int pcolor)
{
if((nx<0)||(nx>XRESO-1)||(ny<0-PPDY)||(ny>YRESO-1)) return 1;
/*if((nx<0)||(nx>XRESO-1)||(ny<0)||(ny>YrESO-1)) return 1;*/

ny+=PPDY;
XSetForeground(d,gcdisplay,irgb[pcolor].pixel);
```

```
XDrawPoint(d,w,gcdisplay,nx,ny);
if(1){
XDrawPoint(d,pmap1,gcdisplay,nx,ny);
if(fieldflag==0 && c_trans==0){
dsp[nx][ny].cx=xi;dsp[nx][ny].cy=yi;  /* nx, ny:display coordinates */
}
}
else{
/*SetPixelV(hdctmp1,nx,ny,PALETTE(pcolor));
if(xg==0){
SetPixelV(hdctmp1,nx+1,ny,PALETTE(pcolor));
SetPixelV(hdctmp1,nx-1,ny,PALETTE(pcolor));
}
else{
SetPixelV(hdctmp1,nx,ny+1,PALETTE(pcolor));
SetPixelV(hdctmp1,nx,ny-1,PALETTE(pcolor));
}*/
}

return 0;
}/** ppixel **/
#endif


void hline(int left,int right,int y,int color)
{
int i,xs,ys;
dbl DX,DY,Dz,X,Y,Z,len,val;

DX=work[right][y].x-work[left][y].x;
DY=work[right][y].y-work[left][y].y;
Dz=work[right][y].z-work[left][y].z;

for(i=left+1;i<=right-1;i++){
xs=i+dO[0]-Zx/2;
ys=y+dO[1]-Zy/2;

if(/*Zflag==0*/1) ppixel(xs,ys,color);
else{
}
}/**for(i)**/
}/** hline **/


void line(dbl x1_,dbl y1_,dbl x2_,dbl y2_,int color)
```

```
{
int x1,y1,x2,y2,dx,dy,x,y;
int c,d,e,sx,sy;
int putflag=1,putcount=0;

x1=ff_fc(x1_);y1=ff_fc(y1_);
x2=ff_fc(x2_);y2=ff_fc(y2_);

dx=abs(x2-x1);
dy=abs(y2-y1);

if(x1<=x2) sx=1;
else       sx=-1;
if(y1<=y2) sy=1;
else       sy=-1;

if(dx>=dy){
c=2*dy;d=2*(dy-dx);e=c-dx;

x=x1;
y=y1;

while(1){
if(putperiod==0) ;
else{
if(putcount==putperiod){
putcount=0;
if(putflag==1) putflag=0;else putflag=1;
}
putcount++;
}

if(putperiod==0 || putflag==1) ppixel(x,y,color);

if(e<0) e+=c;
else    {e+=d;y+=sy;}

x+=sx;
if(sx>=0) {if(x>x2) break;}
else      {if(x<x2) break;}
}
}/**if(dx,dy)**/
else{
c=2*dx;d=2*(dx-dy);e=c-dy;

x=x1;
```

```
y=y1;

while(1){
if(putperiod==0) ;
else{
if(putcount==putperiod){
putcount=0;
if(putflag==1) putflag=0;else putflag=1;
}
putcount++;
}

if(putperiod==0 || putflag==1) ppixel(x,y,color);

if(e<0) e+=c;
else    {e+=d;x+=sx;}

y+=sy;
if(sy>=0) {if(y>y2) break;}
else      {if(y<y2) break;}
}
}/**else(dx,dy)**/
}/** line **/


void line_eye_(dbl xd1,dbl yd1,dbl zd1,dbl xd2,dbl yd2,dbl zd2,int color)
{
int x1,y1,x2,y2,dx,dy,x,y;
int c,d,e,sx,sy;
int putflag=1,putcount=0;
int xb,yb;
dbl DX,DY,Dz,X,Y,Z,len,val,tmp0,tmp1,tmp2;

/*projection(xd1,yd1,zd1,&x1,&y1);
projection(xd2,yd2,zd2,&x2,&y2);*/
x1=xd1;y1=yd1;
x2=xd2;y2=yd2;

dx=x2-x1;
if(dx<0){
dx=x1;dy=y1;
x1=x2;y1=y2;
x2=dx;y2=dy;

tmp0=xd1;tmp1=yd1;tmp2=zd1;
xd1=xd2;yd1=yd2;zd1=zd2;
```

```
xd2=tmp0;yd2=tmp1;zd2=tmp2;
}

DX=xd2-xd1;
DY=yd2-yd1;
Dz=zd2-zd1;




dx=abs(x2-x1);
dy=abs(y2-y1);

if(dx==0 && dy==0){
/* x1,y1 */
if(Zflag==0) ppixel(x1,y1,color);
else{
xb=x1-dO[0]+Zx/2;
yb=y1-dO[1]+Zy/2;
if(xb<0 || xb>Zx-1 || yb<0 || yb>Zy-1) ;
else{
if(/*len<Zbuf[Zpage][xb][yb] || Zflag==2*/1){
/*if(color==0){*/
ppixel(x1,y1,color);
/*}
else{
if(work_Rect[xb][yb]==0) {ppixel(x1,y1,color);work_Rect[xb][yb]=2;}
}*/
}
/*else if(color==0 && work_Rect[xb][yb]==2) ppixel(x1,y1,color);*/

stack[idx].x=xb;stack[idx].y=yb;idx++;
/*if(color==0) work_Rect[xb][yb]=1;
stack_Rect[idx_Rect].x=xb;stack_Rect[idx_Rect].y=yb;idx_Rect++;*/
work[xb][yb].p=1;
work[xb][yb].x=X;
work[xb][yb].y=Y;
work[xb][yb].z=Z;
}
}

return;
}

if(x1<=x2) sx=1;
else       sx=-1;
if(y1<=y2) sy=1;
```

```
else       sy=-1;

x=x1;
y=y1;

if(dx>=dy){
c=2*dy;d=2*(dy-dx);e=c-dx;

while(1){
if(putperiod==0) ;
else{
if(putcount==putperiod){
putcount=0;
if(putflag==1) putflag=0;else putflag=1;
}
putcount++;
}

if(putperiod==0 || putflag==1) ;else goto next_dx;



if(Zflag==0) ppixel(x,y,color);
else{
xb=x-dO[0]+Zx/2;
yb=y-dO[1]+Zy/2;
if(xb<0 || xb>Zx-1 || yb<0 || yb>Zy-1) ;
else{
if(/*len<Zbuf[Zpage][xb][yb] || Zflag==2*/1){
/*if(color==0){*/
ppixel(x,y,color);
/*}
else{
if(work_Rect[xb][yb]==0) {ppixel(x,y,color);work_Rect[xb][yb]=2;}
}*/
}
/*else if(color==0 && work_Rect[xb][yb]==2) ppixel(x,y,color);*/

stack[idx].x=xb;stack[idx].y=yb;idx++;
/*if(color==0) work_Rect[xb][yb]=1;
stack_Rect[idx_Rect].x=xb;stack_Rect[idx_Rect].y=yb;idx_Rect++;*/
work[xb][yb].p=1;
work[xb][yb].x=X;
work[xb][yb].y=Y;
work[xb][yb].z=Z;
}
```

```
}



next_dx:

if(e<0) e+=c;
else    {e+=d;y+=sy;}

x+=sx;
if(sx>=0) {if(x>x2) break;}
else      {if(x<x2) break;}
}
}/**if(dx,dy)**/
else{
c=2*dx;d=2*(dx-dy);e=c-dy;

while(1){
if(putperiod==0) ;
else{
if(putcount==putperiod){
putcount=0;
if(putflag==1) putflag=0;else putflag=1;
}
putcount++;
}

if(putperiod==0 || putflag==1) ;else goto next_dy;



if(Zflag==0) ppixel(x,y,color);
else{
xb=x-dO[0]+Zx/2;
yb=y-dO[1]+Zy/2;
if(xb<0 || xb>Zx-1 || yb<0 || yb>Zy-1) ;
else{
if(/*len<Zbuf[Zpage][xb][yb] || Zflag==2*/1){
/*if(color==0){*/
ppixel(x,y,color);
/*}
else{
if(work_Rect[xb][yb]==0) {ppixel(x,y,color);work_Rect[xb][yb]=2;}
}*/
}
/*else if(color==0 && work_Rect[xb][yb]==2) ppixel(x,y,color);*/
stack[idx].x=xb;stack[idx].y=yb;idx++;
/*if(color==0) work_Rect[xb][yb]=1;
stack_Rect[idx_Rect].x=xb;stack_Rect[idx_Rect].y=yb;idx_Rect++;*/
work[xb][yb].p=1;
work[xb][yb].x=X;
work[xb][yb].y=Y;
work[xb][yb].z=Z;
}
}



next_dy:

if(e<0) e+=c;
else    {e+=d;x+=sx;}

y+=sy;
if(sy>=0) {if(y>y2) break;}
else      {if(y<y2) break;}
}
}/**else(dx,dy)**/
}/** line_eye_ **/


void Trian(char flag,dbl x1,dbl y1,dbl z1,dbl x2,dbl y2,dbl z2,
           dbl x3,dbl y3,dbl z3,int color1,int color2,int color3,int color)
{
int i;
int ymin,ymax,y,xmin,xmax,xmin_,xmax_;

restore_work(0);
/*if(Rectflag==0) restore_work(1);*/

if(flag==0){
line_eye_(x1,y1,-1,x2,y2,-1,color3);
line_eye_(x2,y2,-1,x3,y3,-1,color1);
line_eye_(x3,y3,-1,x1,y1,-1,color2);
}
else{
line_eye_(x1,y1,-1,x2,y2,-1,color3);
line_eye_(x2,y2,-1,x3,y3,-1,color1);
line_eye_(x3,y3,-1,x1,y1,-1,color2);
}
```

```
if(idx==0) return;

i=0;ymin=Zy;
while(1){
if(stack[i].y<ymin) ymin=stack[i].y;

i++;if(i==idx) break;
}

i=0;ymax=-1;
while(1){
if(stack[i].y>ymax) ymax=stack[i].y;

i++;if(i==idx) break;
}

for(y=ymin;y<=ymax;y++){
i=0;xmin=Zx;
while(1){
if(stack[i].y==y && stack[i].x<xmin) xmin=stack[i].x;

i++;if(i==idx) break;
}

i=0;xmax=-1;
while(1){
if(stack[i].y==y && stack[i].x>xmax) xmax=stack[i].x;

i++;if(i==idx) break;
}

i=xmin;
while(1){
if(work[i+1][y].p==0) {xmin_=i;break;}

i++;if(i==Zx-1) break;
}

i=xmax;
while(1){
if(work[i-1][y].p==0) {xmax_=i;break;}

i--;if(i==0) break;
}

if(xmax_>=xmin_+2){
```

```
hline(xmin_,xmax_,y,color);
}
}/**for(y)**/
}/** Trian **/


void Polyline_(POINT *vertex,int num,int pencolor)
{
int i,old;

old=xg;

for(i=0;i<num;i++){
if(vertex[i].x==vertex[i+1].x) xg=0;else xg=1;
line(vertex[i].x,vertex[i].y,vertex[i+1].x,vertex[i+1].y,pencolor);
}

xg=old;
}/** Polyline_ **/


void Polygon_(POINT *vertex,int num,int color)
{
int i,x,y;

/*return;*/

if(num==4){
Trian(0,vertex[0].x,vertex[0].y,-1,vertex[1].x,vertex[1].y,-1,
        vertex[2].x,vertex[2].y,-1,color,color,color,color);
Trian(0,vertex[0].x,vertex[0].y,-1,vertex[2].x,vertex[2].y,-1,
        vertex[3].x,vertex[3].y,-1,color,color,color,color);
}
else if(num==6){
x=(vertex[0].x+vertex[3].x)/2;
y=(vertex[0].y+vertex[3].y)/2;
for(i=0;i<num;i++)
Trian(0,vertex[i].x,vertex[i].y,-1,vertex[i+1].x,vertex[i+1].y,-1,
        x,y,-1,color,color,color,color);
}
}/** Polygon_ **/


void mem_bdr(int x,int y)
{
int i;
```

```
if((i=bdrcheck(x,y))<0){
Fill=1;
bdr[bdrnum].cx=x;bdr[bdrnum].cy=y;
/*search_i(x[0],y[0],-1);*/
/*printf(" %d %d\n",bdr[bdrnum].cx,bdr[bdrnum].cy);*/
bdrnum++;
}
}/** mem_bdr **/


void search_bdr(int x,int y)
{
Fill=1;
bdr[bdrnum].cx=x;bdr[bdrnum].cy=y;
search_i(x,y,-1);
/*printf(" %d %d\n",bdr[bdrnum].cx,bdr[bdrnum].cy);*/
bdrnum++;

mem_bdr(x,y);                     /* c_trans=3, Fill=1 */
}/** search_bdr **/


int getflag(int x,int y)
{
int i,val,dlt=SQSZ;

val=3;

for(i=0;i<0+/*16*/18;i++)
if(x>i*dlt && x<(i+1)*dlt && y>0/*-PPDY*/ && y<dlt/*-PPDY*/){
xg=i-0;
     if(xg==16) val=-1;
else if(xg==17) val=-2;
else            val=0;
break;
}

if(val==3 && Fill==-1)
for(i=/*17*/19;i</*17*/19+5;i++)
if(x>i*dlt && x<(i+1)*dlt && y>0/*-PPDY*/ && y<dlt/*-PPDY*/) {val=1;xg=i-19;break;}

if(val==3)
for(i=/*24*/25;i</*24*/25+4;i++)
if(x>i*dlt && x<(i+1)*dlt && y>0/*-PPDY*/ && y<dlt/*-PPDY*/) {val=2;xg=i-25;break;}
if(val<0){
rectangle_(0,i*dlt,dlt-PPDY+2,(i+1)*dlt,dlt-PPDY+3,bfset[WB].fore,i);
/*BitBlt_full();*/
}
else if(val==0){
rectangle_(0,0*dlt,dlt-PPDY+2,(0+16)*dlt,dlt-PPDY+3,bfset[WB].back,i);  /* erase */
if(i==17){
if(clrpp==0) rectangle_(0,i*dlt,dlt-PPDY+2,(i+1)*dlt,dlt-PPDY+3,bfset[WB].fore,i);
}
else
rectangle_(0,i*dlt,dlt-PPDY+2,(i+1)*dlt,dlt-PPDY+3,bfset[WB].fore,i);
/*BitBlt_full();*/
}
else if(val==1){
rectangle_(0,19*dlt,dlt-PPDY+2,(19+5)*dlt,dlt-PPDY+3,bfset[WB].back,i);  /* erase */
rectangle_(0,i*dlt,dlt-PPDY+2,(i+1)*dlt,dlt-PPDY+3,bfset[WB].fore,i);
/*BitBlt_full();*/
}
else if(val==2){
if(Fill>-1 && xg==0)
rectangle_(0,25*dlt,dlt-PPDY+2,(25+1)*dlt,dlt-PPDY+3,bfset[WB].back,i);  /* erase */
else if(Fill==-1 && xg==0)
rectangle_(0,i*dlt,dlt-PPDY+2,(i+1)*dlt,dlt-PPDY+3,bfset[WB].fore,i);
else if(Fill>-1 && xg==1)
rectangle_(0,i*dlt,dlt-PPDY+2,(i+1)*dlt,dlt-PPDY+3,bfset[WB].fore,i);
else if(Fill>-1 && xg==2)
rectangle_(0,i*dlt,dlt-PPDY+2,(i+1)*dlt,dlt-PPDY+3,bfset[WB].fore,i);
else if(Fill>-1 && xg==3)
rectangle_(0,i*dlt,dlt-PPDY+2,(i+1)*dlt,dlt-PPDY+3,bfset[WB].fore,i);
/*BitBlt_full();*/
}

if(val<0) return 0;
else      return val;
}/** getflag **/


void search_i(int cx,int cy,int pcolor)
{
char flag=0,str[32];
int i,j,k,cnt,cnt_,id_tmp,j_,the_color;

for(i=0;i<RCMAX;i++)
for(j=0;j<CPMAX;j++)
for(k=0;k<10;k++){
if(mbr[i].x[j][k]<0) break;
if(mbr[i].x[j][k]==cx && mbr[i].y[j][k]==cy){
id_tmp=id_1st;
if(1){
/*printf_("id_1st=",id_1st,DX_,0);*/
printf_("ig    =",j,DX_,1);
}

cnt=0;

if(CPHALF==CPMAX){
j_=j;
yg=id_tmp;

while(1){
/* c_trans=3 */
xg=id_tmp;
if(zi==0)
putpixel_(mbr[i].x[j_][0],mbr[i].y[j_][0],pcolor);
else{
the_color=pixel[mbr[i].x[j_][0]][mbr[i].y[j_][0]];
putpixel_(mbr[i].x[j_][0],mbr[i].y[j_][0],the_color);
}
/*printf(" %d %d\n",mbr[i].x[j_][0],mbr[i].y[j_][0]);*/

j_++;if(j_==CPMAX) j_=0;
id_tmp++;if(id_tmp==CPMAX) id_tmp=0;

cnt++;if(cnt==CPMAX) break;
if(clrpp==1 && EDGE==0) {pcolor++;if(pcolor>15) pcolor=0;}
}/**while(1)**/
}/**if(CPHALF)**/
else{
begin:
cnt_=0;
j_=j;
yg=id_tmp;

while(1){
/* c_trans=3 */
xg=id_tmp;
if(zi==0)
putpixel_(mbr[i].x[j_][0],mbr[i].y[j_][0],pcolor);
else{
the_color=pixel[mbr[i].x[j_][0]][mbr[i].y[j_][0]];
putpixel_(mbr[i].x[j_][0],mbr[i].y[j_][0],the_color);
}
/*printf(" %d %d\n",mbr[i].x[j_][0],mbr[i].y[j_][0]);*/

j_++;
if(j<CPHALF){
if(j_==CPHALF) j_=0;
}
else{
if(j_==CPMAX) j_=CPHALF;
}

id_tmp++;if(id_tmp==CPMAX) id_tmp=0;

cnt++;
if(cnt==CPMAX) break;
if(clrpp==1 && EDGE==0) {pcolor++;if(pcolor>15) pcolor=0;}

cnt_++;
if(cnt_==CPHALF){
if(j<CPHALF) j+=CPHALF;
else         j-=CPHALF;
goto begin;
}

}/**while(1)**/
}/**else(CPHALF)**/

flag=1;
/*BitBlt_full();*/
goto end;
}/**if(mbr[][], mbr[][])**/
}

end:
/*printf(" search:%d %d\n",cx,cy)*/;
}/** search_i **/

/************************** dialog functions -> **************************/

#if WX==0
void setcsrcolor(int color)
{
hbrush=CreateSolidBrush(PALETTE(color));
SelectObject(hdctmp3,hbrush);
PatBlt(hdctmp3,0,0,XRESO,/*YRESO*/UDY,PATCOPY);
DeleteObject(hbrush);
```

```
}/** setcsrcolor **/
#else
void setcsrcolor(int color)
{
XSetForeground(d,gcdisplay,irgb[color].pixel);
XFillRectangle(d,pmap3,gcdisplay,0,0,XRESO,UDY);  /* for cursor */
}/** setcsrcolor **/
#endif


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 **/


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


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


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

```
#if WX==0
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();
}
else if(cqflag%2==1) cqflag++;

return;
}

if(usflag==1) return;
/*if(driveflag) return;*/

if(menuflag>0){
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(2);csr();}
else if(BitBltflag_==1)             csr();
else{}

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

myime.ptCurrentPos=point;
ImmSetCompositionWindow(himc,&myime);
}/** InputPosition **/


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

if(menuflag>0){
return;
}/**if(menuflag)*****************************/

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)*******************************/

}/** 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 **/
#else
void WM_func_CHAR(unsigned char charcode_tmp)
{
if(charcode_tmp<0x20 || charcode_tmp>0x7e){
if(cqflag>0 && cqflag%2==0){
/*extraline(1);*/cqflag=0;
if(filerflag) {if(dialogflag>0 && imm_restart_flag==1) imm_restart();}
else imm_restart();
}
else if(cqflag%2==1) cqflag++;

return;
}

if(usflag==1) return;

if(menuflag>0){
return;
}/**if(menuflag)*****************************/

if(dialogflag>0){
charcode=charcode_tmp;

if(cqflag==2){
/*overwrite();
insertion_cc_dialog(charcode);
extraline(1);*/cqflag=0;
if(filerflag) {if(imm_restart_flag==1) imm_restart();}
else imm_restart();
}
else if(cqflag==4 || cqflag==6){    /* 4(<- ex. Esc Q) and 6 */
/*if(noelineflag==0) extraline(1);else noelineflag=0;*/
cqflag=0;
if(filerflag) {if(imm_restart_flag==1) imm_restart();}
else imm_restart();
}
else{
overwrite();
insertion_dialog(charcode);
}

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


char gettype_sdb(long k)
{
char type;
unsigned char s[2];

s[0]=stock_db[k];
/*s[1]=stock_db[k+1];*/

type=gettype(1,s[0]/*,s[1]*/,k,kmax_sdb);

return type;
}/** gettype_sdb **/


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

if(style & XIMPreeditPosition){}
else return;

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

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


void WM_funcIME_CHAR(unsigned char *buf_Xmb)
{
char flag_,function_old,type;
long k,k_sdb;
unsigned char db[2];

strcpy(stock_db,buf_Xmb);
dbsize=strlen(stock_db);
kmax_sdb=dbsize-1;

if(menuflag>0){
return;
}/**if(menuflag)*********************************/

if(dialogflag>0){
dbflag=1;

k_sdb=0;
while(1){
type=gettype_sdb(k_sdb);

if(type==3){
db[0]=stock_db[k_sdb];
db[1]=stock_db[k_sdb+1];

tailcheck_dialog();

flag_=0;

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

/*page_firstk_dialog(firstk_dialog);*/
if(flag_==0){
csr_right_dialog();
}
else{
break;
}

k_sdb+=2;
}/**if(type)**/
else{
if(insertion_dialog(stock_db[k_sdb])==1) break;

k_sdb+=1;
}/**else(type)**/

if(k_sdb>kmax_sdb) break;
}/**while(1)**/

dbflag=0;
page_firstk_dialog(firstk_dialog);

InputPosition(ic,icsr,jcsr);

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


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

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

style=InputStyle(ime);




/*strcpy(fs1,fs2[fontnum]);*/
if(0) strcpy(fs1,fs2[3]);
else if(0){
```

```
/* scalable_ */
h=UDY+/*dh*/0;
h=max(min(h,64),8);
/*printf(" UDX=%d\n",UDX);
printf(" UDY=%d\n",UDY);
printf(" dh=%d\n",dh);
printf(" h=%d\n",h);*/

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

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

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



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



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

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

i=0;
while(1){

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

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

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

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

return 0;
}/** InputStyle **/


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

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

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

return ic;
}/** InputContext **/
#endif


void csr_to_1(void)
{
int dy=0;
```

```
long k;

#if WX==1
XSetFunction(d,gcdisplay,GXxor);
#endif
bitbltflag=1;

if(dialogflag==0){
}/**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)**/

#if WX==1
XSetFunction(d,gcdisplay,GXcopy);
#endif
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(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 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_ **/
```

```
#if WX==0
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_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 **/
#else
void left_keydowns_dialog(void)
{
char gotoflag;
```

```
gotoflag=1;

if(cqflag==6){
if(GKS('S')<0 || GKS('s')<0){
  csr_row_home_dialog();}
else if(GKS('D')<0 || GKS('d')<0){
  csr_row_end_dialog();}

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

if(/*cqflag==6*/gotoflag==-1){
  if(GKS(XK_F12)<0 || GKS(XK_Pause)<0 || GKS(XK_F1)<0) cqflag=0;  /* Esc+ */
  /*BitBltflag_=2;*/
  goto end_lk_dialog;}

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

if(GKS(XK_Delete)<0){
  deletion_dialog();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)>=0 && GKS(XK_BackSpace)<0){
  backspace_dialog();}

else if(GKS(XK_Up)<0){
  restore_dialog();}
else if(GKS(XK_Down)<0){
  clear_dialog(1);}
else if(GKS_(ControlMask)>=0 && GKS(XK_Left)<0){
  csr_left_dialog();}
else if(GKS_(ControlMask)>=0 && GKS(XK_Right)<0){
  csr_right_dialog();}

else if(GKS(XK_Home)<0 || (GKS_(ControlMask)<0 && GKS(XK_Left)<0)){
  csr_row_home_dialog();}
else if(GKS(XK_End)<0 || (GKS_(ControlMask)<0 && GKS(XK_Right)<0)){
  csr_row_end_dialog();}

else if(GKS(XK_Prior)<0){
  page_up_dialog();}
else if(GKS(XK_Next)<0){
  page_down_dialog();}

else if(GKS_(ControlMask)<0 && GKS(XK_Insert)<0){
  if(insorover==0) insorover=1;else insorover=0;
  /*extraline(1);*/BitBltflag_=2;}

/*else if(GKS_(ShiftMask)<0 && GKS_(ControlMask)<0 && (GKS('I')<0 || GKS('i')<0)){
  cqflag=1;prompt_cq(0);}
else if(GKS_(ControlMask)<0 && (GKS('Q')<0 || GKS('q')<0)){
  cqflag=5;prompt_cq(2);}*/
else if(GKS(XK_Tab)<0){
  overwrite();
  insertion_dialog(0x09);}

else if(GKS_(ControlMask)<0 && (GKS('G')<0 || GKS('g')<0)){
  deletion_dialog();}
else if(GKS_(ShiftMask)>=0 && GKS_(ControlMask)<0 && (GKS('H')<0 || GKS('h')<0)){
  backspace_dialog();}

else if(GKS_(ControlMask)<0 && (GKS('E')<0 || GKS('e')<0)){
  restore_dialog();}
else if(GKS_(ControlMask)<0 && (GKS('X')<0 || GKS('x')<0)){
  clear_dialog(1);}
else if(GKS_(ControlMask)<0 && (GKS('S')<0 || GKS('s')<0)){
  csr_left_dialog();}
else if(GKS_(ControlMask)<0 && (GKS('D')<0 || GKS('d')<0)){
  csr_right_dialog();}

else if(GKS_(ControlMask)<0 && (GKS('R')<0 || GKS('r')<0)){
  page_up_dialog();}
else if(GKS_(ControlMask)<0 && (GKS('C')<0 || GKS('c')<0)){
  page_down_dialog();}

else if(GKS_(ControlMask)<0 && (GKS('M')<0 || GKS('m')<0)){
  trim_dialog();
  /*if(GKS_(ShiftMask)<0) use_selector_flag=1;else use_selector_flag=0;*/
  dialogflag=2;refill=0;BitBltflag_=2;}

else {BitBltflag_=2;}

end_lk_dialog:{}
}/** left_keydowns_dialog **/
#endif



#if WX==0
void mainroop(void)
{
MSG msg;
```

```
while(GetMessage(&msg,NULL,0,0)){
TranslateMessage(&msg);
DispatchMessage(&msg);
if(refill!=1) break;
}/**while(GetMessage)**/
}/** mainroop **/
#else
void mainroop(void)
{
while(1){
kbhit_();
if(refill!=1) break;
}/**while(1)**/
}/** mainroop **/
#endif


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*/2,dx,dy,&str[i],1);

i++;
if(i==length) break;
}
}/** title **/


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*/2,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*/1){
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(void)
{
refill=1;
}/** after_mainroop **/


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 **/
```

```
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 **/


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 **/


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);}*/
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,hdc=2;
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+1+WX*(-1);

if(s[0]>=0x20 && type==0)
stc(hdc,dx,dy,s,1);
/*else if(s[0]=='\n'){
s_[0]=0x0d;
stc(hdc,dx,dy,s_,1);
}*/
/*else if(s[0]==0x1a){*/
else if(k==kmax_dialog){
s_[0]=/*0x0d*/dummy_R;
stc(hdc,dx,dy,s_,1);
}
else if(type==-1){
s_[0]=0x0d;
stc(hdc,dx,dy,s_,1);
}
/*else if(s[0]==0x09){
s_[0]=0x0d;
stc(hdc,dx,dy,s_,1);
}*/
else{
if(s[0]==0x7f) s[0]=0x00;
s_[0]=cc[s[0]];
stc(hdc,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(hdc,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);}*/
}/** 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*/2,dx,dy,UDX*(CD+1),UDY);
while_puts_show_dialog(firstk_dialog);

BitBlt_dialog(2);
}/** page_firstk_dialog **/


void BitBlt_dialog(char hdc)
{
int i,j,dx,dy;

i=DI_d-1;j=DJ_d-1;dx=i*UDX;dy=j*UDY;    /* large */
bitblt(hdc,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 fsave(char *str)
{
int i,j;
static char fname[ASIZE]="test.bin";

dialogflag=1;

if(strlen(str)==0){
begin:
strcpy(p_dialog,fname);
before_mainroop("Save");
mainroop();                          /* p_dialog */
after_mainroop();
if(dialogflag==3) goto end;
strcpy(fname,p_dialog);

if((fp=fopen(fname,"w+b"))==NULL){
printf(" Reinput a filename\n");
strcpy(fname,"");goto begin;}
}
else{
fp=fopen(str,"w+b");
}

xy.xt1=3*(RESO-1);
xy.xt2=-9*(RESO-1);
xy.yt=4*(RESO-1);
xy.CPM=CPMAX;
xy.RCM=RCMAX;

fwrite(&xy,12,1,fp);
for(j=0;j<yt+2;j++)
for(i=0;i<xt+2;i++)
fwrite(&pixel[i][j],1,1,fp);

fclose(fp);
printf(" Save\n");

end:
BitBlt_dialog(1);
dialogflag=0;
}/** fsave **/
```

```
void fload(char *str)
{
char val;
int old,i,j;
static char fname[ASIZE]="test.bin";

dialogflag=1;

if(strlen(str)==0){
begin:
strcpy(p_dialog,fname);
before_mainroop("Load");
mainroop();                         /* p_dialog */
after_mainroop();
if(dialogflag==3) goto end;
strcpy(fname,p_dialog);

if((fp=fopen(fname,"r+b"))==NULL){
printf(" Reinput a filename\n");
strcpy(fname,"");goto begin;}
}
else{
fp=fopen(str,"r+b");
}

fread(&xy,12,1,fp);
/*printf(" %d %d %d\n",xy.xt1,xy.xt2,xy.yt);*/

old=EDGE;EDGE=1;

/* 16 */
for(j=0;j<yt+2;j++)
for(i=0;i<xt+2;i++){
fread(&val,1,1,fp);
pixel[i][j]=val;
/*     if(val==-1) {xg=-1;putpixel(i,j,val);}
else */if(val==16) {xg=-1;putpixel(i,j,val);}
/*else              id[i][j]=-2;*/
}

fseek(fp,12,0);
d_trans=1;

/* -1 */
for(j=0;j<yt+2;j++)
for(i=0;i<xt+2;i++){
fread(&val,1,1,fp);
pixel[i][j]=val;
     if(val==-1) {xg=-1;putpixel(i,j,val);}
/*else if(val==16) {xg=-1;putpixel(i,j,val);}*/
/*else              id[i][j]=-2;*/
}

d_trans=0;
fseek(fp,12,0);
EDGE=0;

/* 0~15 */
for(j=0;j<yt+2;j++)
for(i=0;i<xt+2;i++){
fread(&val,1,1,fp);
pixel[i][j]=val;
     if(val==16 || val==-1) {}
else if(val!=-2) {xg=-1;putpixel(i,j,val);}
/*else              id[i][j]=-2;*/
}

EDGE=old;
fclose(fp);
printf(" Load\n");

end:
BitBlt_dialog(1);
dialogflag=0;
}/** fload **/


void restore_edge(void)
{
char val;
int old,i,j;

old=EDGE;EDGE=1;
d_trans=1;

/* -1 */
for(j=0;j<yt+2;j++)
for(i=0;i<xt+2;i++){
val=pixel[i][j];
     if(val==-1) {xg=-1;putpixel(i,j,val);}
/*else if(val==16) {xg=-1;putpixel(i,j,val);}*/
```

```
/*else              id[i][j]=-2;*/
}

d_trans=0;
EDGE=0;

/* 0~15 */
for(j=0;j<yt+2;j++)
for(i=0;i<xt+2;i++){
val=pixel[i][j];
     if(val==16 || val==-1) {}
else if(val!=-2) {xg=id[i][j];putpixel(i,j,val);}
/*else              id[i][j]=-2;*/
}

EDGE=old;
}/** restore_edge **/


int check_id(int num,int id_S,int cx,int cy)
{
int c1,c2,c3,c4,c5,c7;

c1=id[cx+1][cy];
c2=id[cx][cy+1];
if(cx>=1) c3=id[cx-1][cy];else c3=-2;
if(cy>=1) c4=id[cx][cy-1];else c4=-2;

if(num==4){
if(id_S>=0 && c1>=0 && c1!=id_S) return 1;
if(id_S>=0 && c2>=0 && c2!=id_S) return 1;
if(id_S>=0 && c3>=0 && c3!=id_S) return 1;
if(id_S>=0 && c4>=0 && c4!=id_S) return 1;
/* -2:wall pixel */
/*if(c1==-2 || c2==-2 || c3==-2 || c4==-2) return 2;*/
}
else{
c5=id[cx+1][cy+1];
if(cx>=1 && cy>=1) c7=id[cx-1][cy-1];else c7=-2;

if(id_S>=0 && c1>=0 && c1!=id_S) return 1;
if(id_S>=0 && c2>=0 && c2!=id_S) return 1;
if(id_S>=0 && c3>=0 && c3!=id_S) return 1;
if(id_S>=0 && c4>=0 && c4!=id_S) return 1;
if(id_S>=0 && c5>=0 && c5!=id_S) return 1;
if(id_S>=0 && c7>=0 && c7!=id_S) return 1;
```

```
/* -2:wall pixel */
/*if(c1==-2 || c2==-2 || c3==-2 || c4==-2 || c5==-2 || c7==-2) return 2;*/
}

return 0;
}/** check_id **/


int putpixel(int cx,int cy,int pcolor)
{
char flag_id;
int k,dx[2],dy[2],n=RESO,pencolor,num;
POINT vertex[7];

if(cx<0 || cy<0) return 1;
if(!GRPH) goto end;
if(c_trans==1) goto end;
if(c_trans==2 && searchflag==0) goto end;

if(cx>=6*(n-1) && cy<=1*(n-1)){
dx[0]=X0+2*(n-1);
dy[0]=Y0;
if(EDGE==0) {dx[1]=-1;dy[1]=-1;}
else        {dx[1]=0; dy[1]=0;}

vertex[0].x=dx[0]+cx*PIXSIZE;          vertex[0].y=dy[0]+cy*PIXSIZE;
vertex[1].x=dx[0]+(cx+1)*PIXSIZE+dx[1];vertex[1].y=dy[0]+cy*PIXSIZE;
vertex[2].x=dx[0]+(cx+1)*PIXSIZE+dx[1];vertex[2].y=dy[0]+(cy+1)*PIXSIZE+dy[1];
vertex[3].x=dx[0]+cx*PIXSIZE;          vertex[3].y=dy[0]+(cy+1)*PIXSIZE+dy[1];
vertex[4].x=vertex[0].x;               vertex[4].y=vertex[0].y;

num=4;
}
else{
dx[0]=get_dx_h(cx,cy);
dy[0]=get_dy_h(cx,cy);

vertex[0].x=dx[0];                vertex[0].y=dy[0];
vertex[2].x=get_dx_h(cx+1,cy+0);vertex[2].y=get_dy_h(cx+1,cy+0);
vertex[4].x=get_dx_h(cx+1,cy+1);vertex[4].y=get_dy_h(cx+1,cy+1);

vertex[1].x=vertex[4].x      ;vertex[1].y=vertex[0].y-dy_hex;
vertex[3].x=vertex[2].x      ;vertex[3].y=vertex[4].y-dy_hex;
vertex[5].x=vertex[0].x      ;vertex[5].y=vertex[4].y-dy_hex;

vertex[6].x=vertex[0].x;
```

```
vertex[6].y=vertex[0].y;

/*for(i=0;i<7;i++) vertex[i].x-=N1*PIXSIZE*1+15;*/

num=6;
}

if(EDGE==1){
     if(pcolor==15 || pcolor==16){
            pencolor=10;
}
else if(pcolor==0 || pcolor==-1){
if(d_trans) pencolor=bfset[WB].fore;
else        pencolor=15;
}
else        pencolor=15;
}
else{
pencolor=pcolor;
}



if(fieldflag==0){
if(c_trans==0){
}/**if(c_trans)**/
else if(c_trans==1){
}/**else if(c_trans)**/
else if(c_trans==2){
if(searchflag) id[cx][cy]=xg;
flag_id=check_id(num,id[cx][cy],cx,cy);
}/**else if(c_trans)**/
else{
if(Fill==-1){
/* before check_id() */
if(pcolor!=-1 && pcolor!=16) id[cx][cy]=xg;  /* 0,...,15 */
else                         id[cx][cy]=-1;  /* -1, 16 */
}
flag_id=check_id(num,id[cx][cy],cx,cy);
}/**else(c_trans)**/
}/**if(fieldflag)**/
else{
}/**else(fieldflag)**/
```

```
xi=cx;yi=cy;                        /* cx, cy:cag coordinates */

if(EDGE==1){
     if(pcolor==15 || pcolor==16){
            k=15;
}
else if(pcolor==0 || pcolor==-1){
if(d_trans) k=bfset[WB].back;
else        k=0;
}
else        k=pcolor;
Polygon_(vertex,num,k);
Polyline_(vertex,num,pencolor);
}
else if(c_trans==3 && Fill>/*-1*/0){
Polygon_(vertex,num,0);
Polyline_(vertex,num,8);
putperiod=1;
if(xg==yg)
Polyline_(vertex,num,13);
else
Polyline_(vertex,num,14);
putperiod=0;
}
else if((c_trans==3 || (c_trans==2 && searchflag==1)) && flag_id==1){
Polygon_(vertex,num,pcolor);
Polyline_(vertex,num,0);
putperiod=1;
Polyline_(vertex,num,15);
putperiod=0;
}
else if(1){
/*xi=cx;yi=cy;*/                        /* cx, cy:cag coordinates */
Polygon_(vertex,num,pcolor);
Polyline_(vertex,num,pcolor);
}

end:
if(c_trans==0){
if(cx<=xt && cy<=yt) pixel[cx][cy]=pcolor;
}
if(c_trans==3){
if(Fill==-1 && cx<=xt && cy<=yt) pixel[cx][cy]=pcolor;
}
else if(c_trans==1){
if(cx<=xt && cy<=yt) pixel_[cx][cy]=pcolor;
}
else if(c_trans==2){
if(cx<=xt && cy<=yt) {pixel_[cx][cy]=pcolor;pixel[cx][cy]=pcolor;}
}

#if YOUR_ART==1
if(fieldflag==0){
if(c_trans==0){
k=0;
while(1){
if(mbr[rcount[CPMAX-1]].x[ig][k]==-1){
mbr[rcount[CPMAX-1]].x[ig][k]=cx;
mbr[rcount[CPMAX-1]].y[ig][k]=cy;
/*if(k>d_trans) d_trans=k;*/
break;
}
else{
k++;
if(k>9) {printf(" k?\n");refill=0;break;}
}
}/**while(1)**/
}/**if(c_trans)**/
else if(c_trans==1){
}/**else if(c_trans)**/
else if(c_trans==2){
}/**else if(c_trans)**/
else{
/*id[cx][cy]=xg;*/                  /* => before check_id() */
/*printf(" %d\n",xg);*/
}/**else(c_trans)**/
}/**if(fieldflag)**/
else{
if(c_trans==0){
if(pcolor!=0) id[cx][cy]=-1;
else          id[cx][cy]=-2;
}
else if(c_trans==3){
id[cx][cy]=-1;
}
}/**else(fieldflag)**/
#endif

return 0;
}/** putpixel **/
```

```
void check_rcount(void)
{
int i;
long val[2];

if(GRPH>0){
for(i=0;i<CPMAX;i++)
printf(" %ld %ld\n",cnt,rcount[i]);
}
else if(1){
if(/*Odd>0 || CPMAX==5*/1){
val[0]=rcount[0];
for(i=1;i<CPMAX;i++){
if(rcount[i]!=val[0]) {beep(1000);refill=0;break;}
}

if(refill==0)
printf(" %ld %ld %d:%ld\n",cnt,val[0],i,rcount[i]);
else
printf(" %ld %ld\n",cnt,val[0]);
}/**if(_6dRow, CPMAX)**/
else{
val[0]=rcount[0];
for(i=1;i<CPHALF;i++){
if(rcount[i]!=val[0]) {beep(1000);refill=0;break;}
}

if(refill){
val[1]=rcount[CPHALF];
for(i=CPHALF+1;i<CPMAX;i++){
if(rcount[i]!=val[1]) {beep(1000);refill=0;break;}
}

if(refill)
printf(" %ld %ld %ld\n",cnt,val[0],val[1]);
else
printf(" %ld 1st:%ld 2nd:%ld %d:%ld\n",cnt,val[0],val[1],i,rcount[i]);
}/**if(refill)**/
else{
printf(" %ld 1st:%ld %d:%ld\n",cnt,val[0],i,rcount[i]);
}/**else(refill)**/
}/**else(_6dRow, CPMAX)**/
}
else{
printf(" %ld %ld %ld\n",cnt,rcount[0],rcount[1]);
if(rcount[0]!=rcount[1]) {beep(100);refill=0;}
}

if(GRPH==0 && cnt==GRPH_0_MAX) {beep(100);refill=0;}
}/** check_rcount **/


void putdelta(int n,int lr,int x,int y,int color)
{
int i,j;

if(lr==0){
for(j=0;j<=n-1;j++)
for(i=0;i<=j;i++){
if(RTC!=0) putpixel_(x+i,y+j,color);
else{
if(pixel[x+i][y+j]!=16) putpixel_(x+i,y+j,/*fcolor*/16);
}
}
}
else{
for(j=0;j<=n-1;j++)
for(i=0;i<=j;i++){
if(RTC!=0) putpixel_(x+(n-1)-i,y+(n-1)-j,color);
else{
if(pixel[x+(n-1)-i][y+(n-1)-j]!=16) putpixel_(x+(n-1)-i,y+(n-1)-j,/*fcolor*/16);
}
}
}
}/** putdelta **/


void field_s(int x,int y,int color)
{
int i,j,n=RESO;

for(j=y;j<y+n;j++)
for(i=x;i<x+n;i++){
if(RTC!=0) putpixel(i,j,color);
else{
if(pixel[i][j]!=16) putpixel(i,j,/*fcolor*/16);
}
}
}/** field_s **/


void field(int wcolor)
```

```
{
int i,j,n=RESO,fcolor=16,dlt,old;
static int cnt_=0;

fieldflag=1;
old=EDGE;EDGE=1;
if(wcolor==-1) RTC=0;

dlt=0;
/* left_1 */
i=dlt+0;j=0;putdelta(n,1,i*(n-1),j*(n-1),fcolor);
i=dlt+1;j=0;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+1;j=0;putdelta(n,1,i*(n-1),j*(n-1),fcolor);

i=dlt+0;j=0;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+0;j=1;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+0;j=1;putdelta(n,1,i*(n-1),j*(n-1),fcolor);

i=dlt+0;j=2;putdelta(n,1,i*(n-1),j*(n-1),fcolor);
i=dlt+1;j=2;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+1;j=3;putdelta(n,1,i*(n-1),j*(n-1),fcolor);

i=dlt+2;j=3;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+2;j=3;putdelta(n,1,i*(n-1),j*(n-1),fcolor);
i=dlt+3;j=3;putdelta(n,0,i*(n-1),j*(n-1),fcolor);

dlt=3;
/* left_2 */
i=dlt+0;j=0;putdelta(n,1,i*(n-1),j*(n-1),fcolor);
i=dlt+1;j=0;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+1;j=0;putdelta(n,1,i*(n-1),j*(n-1),fcolor);

i=dlt+0;j=0;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+0;j=1;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+0;j=1;putdelta(n,1,i*(n-1),j*(n-1),fcolor);

i=dlt+0;j=2;putdelta(n,1,i*(n-1),j*(n-1),fcolor);
i=dlt+1;j=2;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+1;j=3;putdelta(n,1,i*(n-1),j*(n-1),fcolor);

i=dlt+2;j=3;putdelta(n,0,i*(n-1),j*(n-1),fcolor);
i=dlt+2;j=3;putdelta(n,1,i*(n-1),j*(n-1),fcolor);
i=dlt+3;j=3;putdelta(n,0,i*(n-1),j*(n-1),fcolor);

/* right_1 */
field_s(6*(n-1),0,fcolor);
```

```
/* right_2 */
field_s(9*(n-1),0,fcolor);

if(wcolor==-1) RTC=9;

if(YOUR_ART==1 && !cnt_){
dlt=SQSZ;
for(i=0;i<16;i++)
rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,i);

i=16;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
i=17;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);

i=19;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
i=20;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
i=21;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
i=22;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
i=23;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
setstccolor(0);
i=16;stc(1,i*dlt+9,0.27*dlt-WX*1,"R",1);
i=17;stc(1,i*dlt+6,0.27*dlt-WX*1,"PP",2);

i=19;stc(1,i*dlt+6,0.27*dlt-WX*1,"ID",2);
i=20;stc(1,i*dlt+9,0.27*dlt-WX*1,"C",1);
i=21;stc(1,i*dlt+6,0.27*dlt-WX*1,"AC",2);
i=22;stc(1,i*dlt+9,0.27*dlt-WX*1,"S",1);
i=23;stc(1,i*dlt+9,0.27*dlt-WX*1,"L",1);
setstccolor(bfset[WB].fore);

i=25;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
i=26;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
i=27;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
i=28;rectangle_(1,i*dlt,0-PPDY,(i+1)*dlt,dlt-PPDY,bfset[WB].fore,7);
setstccolor(0);
i=25;stc(1,i*dlt+9,0.27*dlt-WX*1,"F",1);
i=26;stc(1,i*dlt+9,0.27*dlt-WX*1,"C",1);
i=27;stc(1,i*dlt+6,0.27*dlt-WX*1,"AC",2);
i=28;stc(1,i*dlt+9,0.27*dlt-WX*1,"R",1);
setstccolor(bfset[WB].fore);

i=9;rectangle_(0,i*dlt,dlt-PPDY+2,(i+1)*dlt,dlt-PPDY+3,bfset[WB].fore,i);
printf_("id_1st=",id_1st,DX_,0);

cnt_++;
}
```

```
EDGE=old;
fieldflag=0;
}/** field **/


void rectangle_(char flag,int x1,int y1,int x2,int y2,int color1,int color2)
{
int i,j;

line(x1,y1,x2,y1,color1);
line(x2,y1,x2,y2,color1);
line(x2,y2,x1,y2,color1);
line(x1,y2,x1,y1,color1);

if(flag)
for(j=y1+1;j<y2;j++)
for(i=x1+1;i<x2;i++)
ppixel(i,j,color2);
}/** rectangle_ **/


void rot_h(int pos,int dth)
{
pos=pos%6;
dth=dth%6;

pos+=dth;

if(pos>5) pos-=6;
else if(pos<0) pos+=6;

if(pos==0) {tmp0=1;tmp1=0;}  /* ca1 */
else if(pos==1) {tmp0=1;tmp1=1;}  /* ca5 */
else if(pos==2) {tmp0=0;tmp1=1;}  /* ca2 */
else if(pos==3) {tmp0=-1;tmp1=0;}  /* ca3 */
else if(pos==4) {tmp0=-1;tmp1=-1;}  /* ca7 */
else if(pos==5) {tmp0=0;tmp1=-1;}  /* ca4 */
}/** rot_h **/


void rot(int pos,int dth)
{
pos=pos%4;
dth=dth%4;
```

```
pos+=dth;

if(pos>3) pos-=4;
else if(pos<0) pos+=4;

if(pos==0) {tmp0=1;tmp1=0;}  /* ca1 */
else if(pos==1) {tmp0=0;tmp1=1;}  /* ca2 */
else if(pos==2) {tmp0=-1;tmp1=0;}  /* ca3 */
else if(pos==3) {tmp0=0;tmp1=-1;}  /* ca4 */
}/** rot **/


int nh_s(int x,int y,int nx,int ny,int pos)
{
pos=pos%6;

if(pos==0){
if(nx==x+1 && ny==y) return 1;else return 0;
}
else if(pos==1){
if(nx==x && ny==y+1) return 1;else return 0;
}
else if(pos==2){
if(nx==x-1 && ny==y) return 1;else return 0;
}
else if(pos==3){
if(nx==x && ny==y-1) return 1;else return 0;
}
}/** nh_s **/


int nh_h(int x,int y,int nx,int ny,int pos)
{
pos=pos%6;

if(pos==0){
if(nx==x+1 && ny==y) return 1;else return 0;
}
else if(pos==1){
if(nx==x+1 && ny==y+1) return 1;else return 0;
}
else if(pos==2){
if(nx==x && ny==y+1) return 1;else return 0;
}
else if(pos==3){
if(nx==x-1 && ny==y) return 1;else return 0;
```

```
}
else if(pos==4){
if(nx==x-1 && ny==y-1) return 1;else return 0;
}
else if(pos==5){
if(nx==x && ny==y-1) return 1;else return 0;
}
}/** nh_h **/


int side_s_(int sn,int s,int side,int nx,int ny)
{
int n=RESO;

nx-=sn*(n-1);

if(s==1 && side==0){
if(nx>=6*(n-1)+1 && nx<=7*(n-1)-1 && ny==0*(n-1)) return 1;else return 0;
}
else if(s==1 && side==1){
if(nx==7*(n-1) && ny>=0*(n-1)+1 && ny<=1*(n-1)-1) return 1;else return 0;
}
else if(s==1 && side==2){
if(nx>=6*(n-1)+1 && nx<=7*(n-1)-1 && ny==1*(n-1)) return 1;else return 0;
}
else if(s==1 && side==3){
if(nx==6*(n-1) && ny>=0*(n-1)+1 && ny<=1*(n-1)-1) return 1;else return 0;
}

return 0;
}/** side_s_ **/


int side_d_(int sn,int d,int side,int nx,int ny)
{
int n=RESO;

nx-=sn*(n-1);

if(d==3 && side==1){
if(nx>=1*(n-1)+1 && nx<=2*(n-1)-1 && ny==1*(n-1)) return 1;else return 0;
}
else if(d==11 && side==0){
if(nx>=2*(n-1)+1 && nx<=3*(n-1)-1 && ny==3*(n-1)) return 1;else return 0;
}
else if(d==8 && side==0){
```

```
if(nx>=1*(n-1)+1 && nx<=2*(n-1)-1 && ny==nx+1*(n-1)) return 1;else return 0;
}
else if(d==6 && side==1){
if(nx==1*(n-1) && ny>=1*(n-1)+1 && ny<=2*(n-1)-1) return 1;else return 0;
}

else if(d==4 && side==1){
if(nx==2*(n-1) && ny>=0*(n-1)+1 && ny<=1*(n-1)-1) return 1;else return 0;
}
else if(d==12 && side==0){
if(nx>=3*(n-1)+1 && nx<=4*(n-1)-1 && ny==nx+0*(n-1)) return 1;else return 0;
}

else if(d==4 && side==0){
if(nx>=1*(n-1)+1 && nx<=2*(n-1)-1 && ny==0*(n-1)) return 1;else return 0;
}
else if(d==2 && side==0){
if(nx>=0*(n-1)+1 && nx<=1*(n-1)-1 && ny==0*(n-1)) return 1;else return 0;
}
else if(d==7 && side==2){
if(nx>=0*(n-1)+1 && nx<=1*(n-1)-1 && ny==nx+2*(n-1)) return 1;else return 0;
}
else if(d==9 && side==2){
if(nx>=1*(n-1)+1 && nx<=2*(n-1)-1 && ny==nx+2*(n-1)) return 1;else return 0;
}
else if(d==2 && side==0){
if(nx>=0*(n-1)+1 && nx<=1*(n-1)-1 && ny==0*(n-1)) return 1;else return 0;
}
else if(d==1 && side==2){
if(nx==0*(n-1) && ny>=0*(n-1)+1 && ny<=1*(n-1)-1) return 1;else return 0;
}
else if(d==10 && side==1){
if(nx>=2*(n-1)+1 && nx<=3*(n-1)-1 && ny==4*(n-1)) return 1;else return 0;
}
else if(d==5 && side==2){
if(nx==0*(n-1) && ny>=1*(n-1)+1 && ny<=2*(n-1)-1) return 1;else return 0;
}
else if(d==12 && side==1){
if(nx>=3*(n-1)+1 && nx<=4*(n-1)-1 && ny==4*(n-1)) return 1;else return 0;
}

return 0;
}/** side_d_ **/


int v_s_(int sn,int v,int nx,int ny)
```

```
{
int n=RESO;

nx-=sn*(n-1);

if(v==0){
if(nx==6*(n-1) && ny==0*(n-1)) return 1;else return 0;
}
else if(v==1){
if(nx==7*(n-1) && ny==0*(n-1)) return 1;else return 0;
}
else if(v==2){
if(nx==7*(n-1) && ny==1*(n-1)) return 1;else return 0;
}
else if(v==3){
if(nx==6*(n-1) && ny==1*(n-1)) return 1;else return 0;
}

return 0;
}/** v_s_ **/


int v_d_(int sn,int d,int v,int nx,int ny)
{
int n=RESO;

nx-=sn*(n-1);

if(d==3 && v==2){
if(nx==1*(n-1) && ny==1*(n-1)) return 1;else return 0;
}
else if(d==4 && v==1){
if(nx==2*(n-1) && ny==1*(n-1)) return 1;else return 0;
}
else if(d==11 && v==0){
if(nx==3*(n-1) && ny==3*(n-1)) return 1;else return 0;
}
else if(d==8 && v==1){
if(nx==2*(n-1) && ny==3*(n-1)) return 1;else return 0;
}
else if(d==6 && v==1){
if(nx==1*(n-1) && ny==2*(n-1)) return 1;else return 0;
}

else if(d==2 && v==0){               /* A */
if(nx==1*(n-1) && ny==0*(n-1)) return 1;else return 0;
}
else if(d==1 && v==0){              /* B */
if(nx==0*(n-1) && ny==0*(n-1)) return 1;else return 0;
}
else if(d==5 && v==0){              /* C */
if(nx==0*(n-1) && ny==1*(n-1)) return 1;else return 0;
}
else if(d==7 && v==2){              /* D */
if(nx==0*(n-1) && ny==2*(n-1)) return 1;else return 0;
}
else if(d==9 && v==2){              /* E */
if(nx==1*(n-1) && ny==3*(n-1)) return 1;else return 0;
}
else if(d==10 && v==2){             /* F */
if(nx==2*(n-1) && ny==4*(n-1)) return 1;else return 0;
}
else if(d==12 && v==2){              /* G */
if(nx==3*(n-1) && ny==4*(n-1)) return 1;else return 0;
}
else if(d==12 && v==1){              /* H */
if(nx==4*(n-1) && ny==4*(n-1)) return 1;else return 0;
}
else if(d==4 && v==0){              /* I */
if(nx==2*(n-1) && ny==0*(n-1)) return 1;else return 0;
}

return 0;
}/** v_d_ **/


void pp(int nx,int ny,int pcolor)
{
nx*=RESO-1;ny*=RESO-1;
putpixel(nx,ny,pcolor);
}/** pp **/


void putpixel_(int nx,int ny,int pcolor)
{
int n=RESO,flag,dlt,sn,dsn;

putpixel(nx,ny,pcolor);
if(fieldflag) return;

/*goto end;*/  /* here */
if(0) goto next;
```

```
/*nx-=sn*(n-1);*/

/* inside vtx(square+delta) */
if(v_s_((sn=0),0,nx,ny)==1 || v_d_(sn,3,2,nx,ny)==1){
pp(6+sn,0,pcolor);pp(1+sn,1,pcolor);
}
else if(v_s_((sn=3),0,nx,ny)==1 || v_d_(sn,3,2,nx,ny)==1){
pp(6+sn,0,pcolor);pp(1+sn,1,pcolor);
}
else if(v_s_((sn=0),1,nx,ny)==1 || v_d_(sn,4,1,nx,ny)==1 || v_d_(sn,11,0,nx,ny)==1){
pp(7+sn,0,pcolor);pp(2+sn,1,pcolor);pp(3+sn,3,pcolor);
}
else if(v_s_((sn=3),1,nx,ny)==1 || v_d_(sn,4,1,nx,ny)==1 || v_d_(sn,11,0,nx,ny)==1){
pp(7+sn,0,pcolor);pp(2+sn,1,pcolor);pp(3+sn,3,pcolor);
}
else if(v_s_((sn=0),2,nx,ny)==1 || v_d_(sn,8,1,nx,ny)==1){
pp(7+sn,1,pcolor);pp(2+sn,3,pcolor);
}
else if(v_s_((sn=3),2,nx,ny)==1 || v_d_(sn,8,1,nx,ny)==1){
pp(7+sn,1,pcolor);pp(2+sn,3,pcolor);
}
else if(v_s_((sn=0),3,nx,ny)==1 || v_d_(sn,6,1,nx,ny)==1){
pp(6+sn,1,pcolor);pp(1+sn,2,pcolor);
}
else if(v_s_((sn=3),3,nx,ny)==1 || v_d_(sn,6,1,nx,ny)==1){
pp(6+sn,1,pcolor);pp(1+sn,2,pcolor);
}

/* outside vtx(delta) */
else if(v_d_((0),2,0,nx,ny)==1 || v_d_((3),1,0,nx,ny)==1){  /* A,B' */
pp(1+(0),0,pcolor);pp(0+(3),0,pcolor);
}
else if(v_d_((0),9,2,nx,ny)==1 || v_d_((3),10,2,nx,ny)==1){  /* E,F' */
pp(1+(0),3,pcolor);pp(2+(3),4,pcolor);
}
else if(v_d_((0),7,2,nx,ny)==1 || v_d_((3),9,2,nx,ny)==1){  /* D,E' */
pp(0+(0),2,pcolor);pp(1+(3),3,pcolor);
}
else if(v_d_((0),12,1,nx,ny)==1 || v_d_((0),4,0,nx,ny)==1 || v_d_((3),2,0,nx,ny)==1){
pp(4+(0),4,pcolor);pp(2+(0),0,pcolor);pp(1+(3),0,pcolor);  /* H(I),A' */
}
else if(v_d_((0),5,0,nx,ny)==1 || v_d_((3),7,2,nx,ny)==1){  /* C,D' */
pp(0+(0),1,pcolor);pp(0+(3),2,pcolor);
}
else if(v_d_((0),12,2,nx,ny)==1 || v_d_((3),12,1,nx,ny)==1 || v_d_((3),4,0,nx,ny)==1){
pp(3+(0),4,pcolor);pp(4+(3),4,pcolor);pp(2+(3),0,pcolor);  /* G,H'(I') */
```

```
}
else if(v_d_((0),1,0,nx,ny)==1 || v_d_((3),5,0,nx,ny)==1){  /* B,C' */
pp(0+(0),0,pcolor);pp(0+(3),1,pcolor);
}
else if(v_d_((0),10,2,nx,ny)==1 || v_d_((3),12,2,nx,ny)==1){  /* F,G' */
pp(2+(0),4,pcolor);pp(3+(3),4,pcolor);
}

else{
next:
sn=0;
dsn=0;
/*999*/
/* inside side(square) */
     if(side_s_(0,1,0,nx,ny)==1) {flag=101;}
else if(side_s_(3,1,0,nx,ny)==1) {flag=101;sn=3;}
else if(side_s_(0,1,1,nx,ny)==1) {flag=102;}
else if(side_s_(3,1,1,nx,ny)==1) {flag=102;sn=3;}
else if(side_s_(0,1,2,nx,ny)==1) {flag=103;}
else if(side_s_(3,1,2,nx,ny)==1) {flag=103;sn=3;}
else if(side_s_(0,1,3,nx,ny)==1) {flag=104;}
else if(side_s_(3,1,3,nx,ny)==1) {flag=104;sn=3;}

/* inside side(delta) */
else if(side_d_(0,3,1,nx,ny)==1) {flag=105;}
else if(side_d_(3,3,1,nx,ny)==1) {flag=105;sn=3;}
else if(side_d_(0,11,0,nx,ny)==1) {flag=106;}
else if(side_d_(3,11,0,nx,ny)==1) {flag=106;sn=3;}
else if(side_d_(0,8,0,nx,ny)==1) {flag=107;}
else if(side_d_(3,8,0,nx,ny)==1) {flag=107;sn=3;}
else if(side_d_(0,6,1,nx,ny)==1) {flag=108;}
else if(side_d_(3,6,1,nx,ny)==1) {flag=108;sn=3;}

/* cut */
else if(side_d_(0,4,1,nx,ny)==1) {flag=109;}
else if(side_d_(3,4,1,nx,ny)==1) {flag=109;sn=3;}
else if(side_d_(0,12,0,nx,ny)==1) {flag=110;}
else if(side_d_(3,12,0,nx,ny)==1) {flag=110;sn=3;}

/* outside side */
#if 1
else if(side_d_(0,4,0,nx,ny)==1) {flag=111;     dsn=3;}  /* 4(0) => 2(3) */
/*else if(side_d_(3,4,0,nx,ny)==1) {flag=111;sn=3;dsn=-3;}*/
/*else if(side_d_(0,2,0,nx,ny)==1) {flag=112;     dsn=3;}*/
else if(side_d_(3,2,0,nx,ny)==1) {flag=112;sn=3;dsn=-3;}  /* 2(3) => 4(0) */
else if(side_d_(0,7,2,nx,ny)==1) {flag=113;     dsn=3;}  /* 7(0) => 9(3) */
```

```
/*else if(side_d_(3,7,2,nx,ny)==1) {flag=113;sn=3;dsn=-3;}*/
/*else if(side_d_(0,9,2,nx,ny)==1) {flag=114;     dsn=3;}*/
else if(side_d_(3,9,2,nx,ny)==1) {flag=114;sn=3;dsn=-3;}  /* 9(3) => 7(0) */
#endif

#if 1
else if(side_d_(0,2,0,nx,ny)==1) {flag=115;     dsn=3;}  /* 2(0) => 1(3) */
/*else if(side_d_(3,2,0,nx,ny)==1) {flag=115;sn=3;dsn=-3;}*/
/*else if(side_d_(0,1,2,nx,ny)==1) {flag=116;     dsn=3;}*/
else if(side_d_(3,1,2,nx,ny)==1) {flag=116;sn=3;dsn=-3;}  /* 1(3) => 2(0) */
else if(side_d_(0,9,2,nx,ny)==1) {flag=117;     dsn=3;}  /* 9(0) => 10(3) */
/*else if(side_d_(3,9,2,nx,ny)==1) {flag=117;sn=3;dsn=-3;}*/
/*else if(side_d_(0,10,1,nx,ny)==1) {flag=118;     dsn=3;}*/
else if(side_d_(3,10,1,nx,ny)==1) {flag=118;sn=3;dsn=-3;}  /* 10(3) => 9(0) */
#endif

#if 1
else if(side_d_(0,1,2,nx,ny)==1) {flag=119;     dsn=3;}  /* 1(0) => 5(3) */
/*else if(side_d_(3,1,2,nx,ny)==1) {flag=119;sn=3;dsn=-3;}*/
/*else if(side_d_(0,5,2,nx,ny)==1) {flag=120;     dsn=3;}*/
else if(side_d_(3,5,2,nx,ny)==1) {flag=120;sn=3;dsn=-3;}  /* 5(3) => 1(0) */
else if(side_d_(0,10,1,nx,ny)==1) {flag=121;     dsn=3;}  /* 10(0) => 12(3) */
/*else if(side_d_(3,10,1,nx,ny)==1) {flag=121;sn=3;dsn=-3;}*/
/*else if(side_d_(0,12,1,nx,ny)==1) {flag=122;     dsn=3;}*/
else if(side_d_(3,12,1,nx,ny)==1) {flag=122;sn=3;dsn=-3;}  /* 12(3) => 10(0) */
#endif

#if 1
else if(side_d_(0,5,2,nx,ny)==1) {flag=123;     dsn=3;}  /* 5(0) => 7(3) */
/*else if(side_d_(3,5,2,nx,ny)==1) {flag=123;sn=3;dsn=-3;}*/
/*else if(side_d_(0,7,2,nx,ny)==1) {flag=124;     dsn=3;}*/  /* 7(3) => 5(0) */
else if(side_d_(3,7,2,nx,ny)==1) {flag=124;sn=3;dsn=-3;}
else if(side_d_(0,12,1,nx,ny)==1) {flag=125;     dsn=3;}  /* 12(0) => 4(3) */
/*else if(side_d_(3,12,1,nx,ny)==1) {flag=125;sn=3;dsn=-3;}*/
/*else if(side_d_(0,4,0,nx,ny)==1) {flag=126;     dsn=3;}*/
else if(side_d_(3,4,0,nx,ny)==1) {flag=126;sn=3;dsn=-3;}  /* 4(3) => 12(0) */
#endif

else flag=-1;

if(flag>0){
nx-=sn*(n-1);

if(flag==101){
dlt=nx-6*(n-1);nx=1*(n-1)+dlt;ny=1*(n-1);;
}
```

```
else if(flag==102){
dlt=1*(n-1)-ny;nx=2*(n-1)+dlt;ny=3*(n-1);;
}
else if(flag==103){
dlt=nx-6*(n-1);nx=1*(n-1)+dlt;ny=2*(n-1)+dlt;;
}
else if(flag==104){
dlt=ny-0*(n-1);nx=1*(n-1);ny=1*(n-1)+dlt;;
}

else if(flag==105){
dlt=nx-1*(n-1);nx=6*(n-1)+dlt;ny=0*(n-1);;
}
else if(flag==106){
dlt=3*(n-1)-nx;nx=7*(n-1);ny=0*(n-1)+dlt;;
}
else if(flag==107){
dlt=nx-1*(n-1);nx=6*(n-1)+dlt;ny=1*(n-1);;
}
else if(flag==108){
dlt=ny-1*(n-1);nx=6*(n-1);ny=0*(n-1)+dlt;;
}

else if(flag==109){
dlt=1*(n-1)-ny;nx=3*(n-1)+dlt;ny=3*(n-1)+dlt;;
}
else if(flag==110){
dlt=4*(n-1)-nx;nx=2*(n-1);ny=0*(n-1)+dlt;;
}

else if(flag==111){
dlt=nx-1*(n-1);nx=0*(n-1)+dlt;ny=0*(n-1);;
}
else if(flag==112){
dlt=nx-0*(n-1);nx=1*(n-1)+dlt;ny=0*(n-1);;
}
else if(flag==113){
dlt=nx-0*(n-1);nx=1*(n-1)+dlt;ny=3*(n-1)+dlt;;
}
else if(flag==114){
dlt=nx-1*(n-1);nx=0*(n-1)+dlt;ny=2*(n-1)+dlt;;
}

else if(flag==115){
dlt=nx-0*(n-1);nx=0*(n-1);ny=1*(n-1)-dlt;;
}
```

```
else if(flag==116){
dlt=ny-0*(n-1);nx=1*(n-1)-dlt;ny=0*(n-1);;
}
else if(flag==117){
dlt=nx-1*(n-1);nx=2*(n-1)+dlt;ny=4*(n-1);;
}
else if(flag==118){
dlt=nx-2*(n-1);nx=1*(n-1)+dlt;ny=3*(n-1)+dlt;;
}

else if(flag==119){
dlt=ny-0*(n-1);nx=0*(n-1);ny=1*(n-1)+dlt;;
}
else if(flag==120){
dlt=ny-1*(n-1);nx=0*(n-1);ny=0*(n-1)+dlt;;
}
else if(flag==121){
dlt=nx-2*(n-1);nx=3*(n-1)+dlt;ny=4*(n-1);;
}
else if(flag==122){
dlt=nx-3*(n-1);nx=2*(n-1)+dlt;ny=4*(n-1);;
}

else if(flag==123){
dlt=ny-1*(n-1);nx=0*(n-1)+dlt;ny=2*(n-1)+dlt;;
}
else if(flag==124){
dlt=nx-0*(n-1);nx=0*(n-1);ny=1*(n-1)+dlt;;
}
else if(flag==125){
dlt=nx-3*(n-1);nx=2*(n-1)-dlt;ny=0*(n-1);;
}
else if(flag==126){
dlt=nx-1*(n-1);nx=4*(n-1)-dlt;ny=4*(n-1);;
}

nx+=sn*(n-1);
nx+=dsn*(n-1);
putpixel(nx,ny,pcolor);
}/**if(flag>0)**/
}/**else(v_s(), v_h())**/

end:
rcount[ig]++;
}/** putpixel_ **/
```

```
int getpixel_(int x,int y,int nx,int ny)
{
int i,n=RESO,flag,dlt,sn,dsn;

/*x-=sn*(n-1);
nx-=sn*(n-1);*/
sn=0;
dsn=0;

if(x<6*(n-1)){
/*if(nx<0) return 1;*/  /* here:0(old) or 1 */
}
else{
/*if(ny<0) return 1;*/                  /* protection for square */
}

if(0) ;

/* cut */
else if(side_d_(0,4,1,x,y)==1) {flag=109;}
else if(side_d_(3,4,1,x,y)==1) {flag=109;sn=3;}
else if(side_d_(0,12,0,x,y)==1) {flag=110;}
else if(side_d_(3,12,0,x,y)==1) {flag=110;sn=3;}

/* outside side */
#if 1
else if(side_d_(0,4,0,x,y)==1) {flag=111;}  /* 4(0) => 2(3) */
/*else if(side_d_(3,4,0,x,y)==1) {flag=111;sn=3;}*/
/*else if(side_d_(0,2,0,x,y)==1) {flag=112;}*/
else if(side_d_(3,2,0,x,y)==1) {flag=112;sn=3;}  /* 2(3) => 4(0) */
else if(side_d_(0,7,2,x,y)==1) {flag=113;}  /* 7(0) => 9(3) */
/*else if(side_d_(3,7,2,x,y)==1) {flag=113;sn=3;}*/
/*else if(side_d_(0,9,2,x,y)==1) {flag=114;}*/
else if(side_d_(3,9,2,x,y)==1) {flag=114;sn=3;}  /* 9(3) => 7(0) */
#endif

#if 1
else if(side_d_(0,2,0,x,y)==1) {flag=115;}  /* 2(0) => 1(3) */
/*else if(side_d_(3,2,0,x,y)==1) {flag=115;sn=3;}*/
/*else if(side_d_(0,1,2,x,y)==1) {flag=116;}*/
else if(side_d_(3,1,2,x,y)==1) {flag=116;sn=3;}  /* 1(3) => 2(0) */
else if(side_d_(0,9,2,x,y)==1) {flag=117;}  /* 9(0) => 10(3) */
/*else if(side_d_(3,9,2,x,y)==1) {flag=117;sn=3;}*/
/*else if(side_d_(0,10,1,x,y)==1) {flag=118;}*/
else if(side_d_(3,10,1,x,y)==1) {flag=118;sn=3;}  /* 10(3) => 9(0) */
```

```
#endif

#if 1
else if(side_d_(0,1,2,x,y)==1) {flag=119;}  /* 1(0) => 5(3) */
/*else if(side_d_(3,1,2,x,y)==1) {flag=119;sn=3;}*/
/*else if(side_d_(0,5,2,x,y)==1) {flag=120;}*/
else if(side_d_(3,5,2,x,y)==1) {flag=120;sn=3;}  /* 5(3) => 1(0) */
else if(side_d_(0,10,1,x,y)==1) {flag=121;}  /* 10(0) => 12(3) */
/*else if(side_d_(3,10,1,x,y)==1) {flag=121;sn=3;}*/
/*else if(side_d_(0,12,1,x,y)==1) {flag=122;}*/
else if(side_d_(3,12,1,x,y)==1) {flag=122;sn=3;}  /* 12(3) => 10(0) */
#endif

#if 1
else if(side_d_(0,5,2,x,y)==1) {flag=123;}  /* 5(0) => 7(3) */
/*else if(side_d_(3,5,2,x,y)==1) {flag=123;sn=3;}*/
/*else if(side_d_(0,7,2,x,y)==1) {flag=124;}*/  /* 7(3) => 5(0) */
else if(side_d_(3,7,2,x,y)==1) {flag=124;sn=3;}
else if(side_d_(0,12,1,x,y)==1) {flag=125;}  /* 12(0) => 4(3) */
/*else if(side_d_(3,12,1,x,y)==1) {flag=125;sn=3;}*/
/*else if(side_d_(0,4,0,x,y)==1) {flag=126;}*/
else if(side_d_(3,4,0,x,y)==1) {flag=126;sn=3;}  /* 4(3) => 12(0) */
#endif

/*999*/
else flag=0;

X=nx;Y=ny;
X_=x;Y_=y;
jmpflag=0;
/*goto end;*/  /* here -> 0 or 1 in arrayreset() */

if(0) ;

else if(flag==109){  /* side(4,1, */
     if(nh_h(x,y,nx,ny,0)==1) {rot_h(0,2);jmpflag=109;}
else if(nh_h(x,y,nx,ny,1)==1) {rot_h(1,2);jmpflag=109;}
if(jmpflag==109){
x-=sn*(n-1);
dlt=1*(n-1)-y;x=3*(n-1)+dlt;y=3*(n-1)+dlt;
x+=sn*(n-1);
}
}
else if(flag==110){  /* side(12,0, */
     if(nh_h(x,y,nx,ny,5)==1) {rot_h(5,-2);jmpflag=110;}
else if(nh_h(x,y,nx,ny,0)==1) {rot_h(0,-2);jmpflag=110;}
```

```
if(jmpflag==110){
x-=sn*(n-1);
dlt=4*(n-1)-x;x=2*(n-1);y=0*(n-1)+dlt;
x+=sn*(n-1);
}
}

else if(flag==111){  /* side_h(4,0, */
     if(nh_h(x,y,nx,ny,4)==1) {rot_h(4,-2);jmpflag=111;}
else if(nh_h(x,y,nx,ny,5)==1) {rot_h(5,2);jmpflag=111;}
if(jmpflag==111){
x-=sn*(n-1);
dlt=x-1*(n-1);x=0*(n-1)+dlt;y=0*(n-1);;dsn=3;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==112){  /* side_h(2,0, */
     if(nh_h(x,y,nx,ny,4)==1) {rot_h(4,-2);jmpflag=112;}
else if(nh_h(x,y,nx,ny,5)==1) {rot_h(5,2);jmpflag=112;}
if(jmpflag==112){
x-=sn*(n-1);
dlt=x-0*(n-1);x=1*(n-1)+dlt;y=0*(n-1);;dsn=-3;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==113){  /* side_h(7,2, */
     if(nh_h(x,y,nx,ny,2)==1) {rot_h(2,-2);jmpflag=113;}
else if(nh_h(x,y,nx,ny,3)==1) {rot_h(3,2);jmpflag=113;}
if(jmpflag==113){
x-=sn*(n-1);
dlt=x-0*(n-1);x=1*(n-1)+dlt;y=3*(n-1)+dlt;;dsn=3;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==114){  /* side_h(9,2, */
     if(nh_h(x,y,nx,ny,2)==1) {rot_h(2,-2);jmpflag=114;}
else if(nh_h(x,y,nx,ny,3)==1) {rot_h(3,2);jmpflag=114;}
if(jmpflag==114){
x-=sn*(n-1);
dlt=x-1*(n-1);x=0*(n-1)+dlt;y=2*(n-1)+dlt;;dsn=-3;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
```

```
}

else if(flag==115){  /* side_h(2,0, */
     if(nh_h(x,y,nx,ny,4)==1) {rot_h(4,3);jmpflag=115;}
else if(nh_h(x,y,nx,ny,5)==1) {rot_h(5,1);jmpflag=115;}
if(jmpflag==115){
x-=sn*(n-1);
dlt=x-0*(n-1);x=0*(n-1);y=1*(n-1)-dlt;;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==116){  /* side_h(1,2, */
     if(nh_h(x,y,nx,ny,3)==1) {rot_h(3,-1);jmpflag=116;}
else if(nh_h(x,y,nx,ny,4)==1) {rot_h(4,-3);jmpflag=116;}
if(jmpflag==116){
x-=sn*(n-1);
dlt=y-0*(n-1);x=1*(n-1)-dlt;y=0*(n-1);;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==117){  /* side_h(9,2, */
     if(nh_h(x,y,nx,ny,2)==1) {rot_h(2,3);jmpflag=117;}
else if(nh_h(x,y,nx,ny,3)==1) {rot_h(3,1);jmpflag=117;}
if(jmpflag==117){
x-=sn*(n-1);
dlt=x-1*(n-1);x=2*(n-1)+dlt;y=4*(n-1);;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==118){  /* side_h(10,1, */
     if(nh_h(x,y,nx,ny,1)==1) {rot_h(1,-1);jmpflag=118;}
else if(nh_h(x,y,nx,ny,2)==1) {rot_h(2,-3);jmpflag=118;}
if(jmpflag==118){
x-=sn*(n-1);
dlt=x-2*(n-1);x=1*(n-1)+dlt;y=3*(n-1)+dlt;;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}

else if(flag==119){  /* side_h(1,2, */
     if(nh_h(x,y,nx,ny,3)==1) {rot_h(3,-2);jmpflag=119;}
else if(nh_h(x,y,nx,ny,4)==1) {rot_h(4,2);jmpflag=119;}
```

```
if(jmpflag==119){
x-=sn*(n-1);
dlt=y-0*(n-1);x=0*(n-1);y=1*(n-1)+dlt;;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==120){  /* side_h(5,2, */
     if(nh_h(x,y,nx,ny,3)==1) {rot_h(3,-2);jmpflag=120;}
else if(nh_h(x,y,nx,ny,4)==1) {rot_h(4,2);jmpflag=120;}
if(jmpflag==120){
x-=sn*(n-1);
dlt=y-1*(n-1);x=0*(n-1);y=0*(n-1)+dlt;;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==121){  /* side_h(10,1, */
     if(nh_h(x,y,nx,ny,1)==1) {rot_h(1,-2);jmpflag=121;}
else if(nh_h(x,y,nx,ny,2)==1) {rot_h(2,2);jmpflag=121;}
if(jmpflag==121){
x-=sn*(n-1);
dlt=x-2*(n-1);x=3*(n-1)+dlt;y=4*(n-1);;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==122){  /* side_h(12,1, */
     if(nh_h(x,y,nx,ny,1)==1) {rot_h(1,-2);jmpflag=122;}
else if(nh_h(x,y,nx,ny,2)==1) {rot_h(2,2);jmpflag=122;}
if(jmpflag==122){
x-=sn*(n-1);
dlt=x-3*(n-1);x=2*(n-1)+dlt;y=4*(n-1);;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}

else if(flag==123){  /* side_h(5,2, */
     if(nh_h(x,y,nx,ny,3)==1) {rot_h(3,-3);jmpflag=123;}
else if(nh_h(x,y,nx,ny,4)==1) {rot_h(4,1);jmpflag=123;}
if(jmpflag==123){
x-=sn*(n-1);
dlt=y-1*(n-1);x=0*(n-1)+dlt;y=2*(n-1)+dlt;;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
```

```
}
}
else if(flag==124){  /* side_h(7,2, */
     if(nh_h(x,y,nx,ny,2)==1) {rot_h(2,-1);jmpflag=124;}
else if(nh_h(x,y,nx,ny,3)==1) {rot_h(3,3);jmpflag=124;}
if(jmpflag==124){
x-=sn*(n-1);
dlt=x-0*(n-1);x=0*(n-1);y=1*(n-1)+dlt;;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==125){  /* side_h(12,1, */
     if(nh_h(x,y,nx,ny,1)==1) {rot_h(1,1);jmpflag=125;}
else if(nh_h(x,y,nx,ny,2)==1) {rot_h(2,-1);jmpflag=125;}
if(jmpflag==125){
x-=sn*(n-1);
dlt=x-3*(n-1);x=2*(n-1)-dlt;y=0*(n-1);;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}
else if(flag==126){  /* side_h(12,1, */
     if(nh_h(x,y,nx,ny,4)==1) {rot_h(4,1);jmpflag=126;}
else if(nh_h(x,y,nx,ny,5)==1) {rot_h(5,-1);jmpflag=126;}
if(jmpflag==126){
x-=sn*(n-1);
dlt=x-1*(n-1);x=4*(n-1)-dlt;y=4*(n-1);;
x+=sn*(n-1);
if(sn==0) dsn=3;else dsn=-3;
}
}

if(jmpflag>=109 && jmpflag<=126){
x+=dsn*(n-1);
X=x+tmp0;
Y=y+tmp1;
X_=x;
Y_=y;
}

end:
if(X<0 || Y<0) return 1;            /* protection */

if(c_trans==0) return pixel[X][Y];
else{
```

```
if(!zeroFill)   return pixel_[X][Y];
else            return pixel[X][Y];
}
}/** getpixel_ **/


int random_(int n)
{
int val;

val=(int)((rand()/(RAND_MAX+1.))*n);

return val;
}/** random_ **/


long ftell_mem(int i)
{
return fp_mem[i];
}/** ftell_mem **/


void fwrite_mem(int i)
{
rtn[i][fp_mem[i]]=s;
fp_mem[i]++;if(fp_mem[i]>asize-1) refill=0;
}/** fwrite_mem **/


void fread_mem(int i)
{
fp_mem[i]--;if(fp_mem[i]<0) fp_mem[i]=0;
s=rtn[i][fp_mem[i]];
}/** fread_mem **/


int fen(char *str,int i,int jmax)
{
int val;

if(i==jmax+1) val=0;
else if(i==-1) val=jmax;
else val=i;

if(strcmp(str,"X")==0) return enX[val];
else if(strcmp(str,"Y")==0) return enY[val];
else if(strcmp(str,"X_")==0) return enX_[val];
else if(strcmp(str,"Y_")==0) return enY_[val];

else if(strcmp(str,"SN")==0) return enSN[val];
}/** fen **/


void close_vtx(void)
{
}/** close_vtx **/


int check_v(int Nx,int Ny)
{
int val;

/*Nx-=sn*(RESO-1);*/
sn=0;

if(Nx>=6*(RESO-1) && Ny<=1*(RESO-1)){
/* inside side(square) */
     if(side_s_(0,1,0,Nx,Ny)==1) {val=200;}  /* <=> 101 */
else if(side_s_(3,1,0,Nx,Ny)==1) {val=200;sn=3;}
else if(side_s_(0,1,1,Nx,Ny)==1) {val=201;}
else if(side_s_(3,1,1,Nx,Ny)==1) {val=201;sn=3;}
else if(side_s_(0,1,2,Nx,Ny)==1) {val=202;}
else if(side_s_(3,1,2,Nx,Ny)==1) {val=202;sn=3;}
else if(side_s_(0,1,3,Nx,Ny)==1) {val=203;}
else if(side_s_(3,1,3,Nx,Ny)==1) {val=203;sn=3;}

/* inside vtx(square) */
#if 1
else if(v_s_(0,0,Nx,Ny)==1) {val=250;}
else if(v_s_(3,0,Nx,Ny)==1) {val=250;sn=3;}
else if(v_s_(0,1,Nx,Ny)==1) {val=251;}
else if(v_s_(3,1,Nx,Ny)==1) {val=251;sn=3;}
else if(v_s_(0,2,Nx,Ny)==1) {val=252;}
else if(v_s_(3,2,Nx,Ny)==1) {val=252;sn=3;}
else if(v_s_(0,3,Nx,Ny)==1) {val=253;}
else if(v_s_(3,3,Nx,Ny)==1) {val=253;sn=3;}
#endif

else
val=299;
}
```

```
else{
/* inside side(delta) */
     if(side_d_(0,3,1,Nx,Ny)==1) {val=300;}  /* <=> 105 */
else if(side_d_(3,3,1,Nx,Ny)==1) {val=300;sn=3;}
else if(side_d_(0,11,0,Nx,Ny)==1) {val=301;}
else if(side_d_(3,11,0,Nx,Ny)==1) {val=301;sn=3;}
else if(side_d_(0,8,0,Nx,Ny)==1) {val=302;}
else if(side_d_(3,8,0,Nx,Ny)==1) {val=302;sn=3;}
else if(side_d_(0,6,1,Nx,Ny)==1) {val=303;}
else if(side_d_(3,6,1,Nx,Ny)==1) {val=303;sn=3;}

/* inside vtx(delta) */
#if 1
else if(v_d_(0,3,2,Nx,Ny)==1) {val=350;}
else if(v_d_(3,3,2,Nx,Ny)==1) {val=350;sn=3;}
else if(v_d_(0,4,1,Nx,Ny)==1) {val=351;}
else if(v_d_(3,4,1,Nx,Ny)==1) {val=351;sn=3;}
else if(v_d_(0,11,0,Nx,Ny)==1) {val=352;}
else if(v_d_(3,11,0,Nx,Ny)==1) {val=352;sn=3;}
else if(v_d_(0,8,1,Nx,Ny)==1) {val=353;}
else if(v_d_(3,8,1,Nx,Ny)==1) {val=353;sn=3;}
else if(v_d_(0,6,1,Nx,Ny)==1) {val=354;}
else if(v_d_(3,6,1,Nx,Ny)==1) {val=354;sn=3;}
#endif

/* outside vtx */
#if 1
else if(v_d_(0,2,0,Nx,Ny)==1) {val=355;}  /* A=>B' */
/*else if(v_d_(3,2,0,Nx,Ny)==1) {val=355;sn=3;}*/
/*else if(v_d_(0,1,0,Nx,Ny)==1) {val=356;}*/
else if(v_d_(3,1,0,Nx,Ny)==1) {val=356;sn=3;}  /* B'=>A */
else if(v_d_(0,9,2,Nx,Ny)==1) {val=357;}  /* E=>F' */
/*else if(v_d_(3,9,2,Nx,Ny)==1) {val=357;sn=3;}
/*else if(v_d_(0,10,2,Nx,Ny)==1) {val=358;}*/
else if(v_d_(3,10,2,Nx,Ny)==1) {val=358;sn=3;}  /* F'=>E */
#endif

#if 1
else if(v_d_(0,7,2,Nx,Ny)==1) {val=359;}  /* D=>E' */
/*else if(v_d_(3,7,2,Nx,Ny)==1) {val=359;sn=3;}*/
/*else if(v_d_(0,9,2,Nx,Ny)==1) {val=360;}*/
else if(v_d_(3,9,2,Nx,Ny)==1) {val=360;sn=3;}  /* E'=>D */
else if(v_d_(0,12,1,Nx,Ny)==1) {val=361;}  /* H=>A' */
/*else if(v_d_(3,12,1,Nx,Ny)==1) {val=361;sn=3;}*/
else if(v_d_(0,4,0,Nx,Ny)==1) {val=362;}  /* I=>A' */
/*else if(v_d_(3,4,0,Nx,Ny)==1) {val=362;sn=3;}*/
```

```
/*else if(v_d_(0,2,0,Nx,Ny)==1) {val=363;}*/
else if(v_d_(3,2,0,Nx,Ny)==1) {val=363;sn=3;}  /* A'=>I */
#endif

#if 1
else if(v_d_(0,5,0,Nx,Ny)==1) {val=364;}  /* C=>D' */
/*else if(v_d_(3,5,0,Nx,Ny)==1) {val=364;sn=3;}*/
/*else if(v_d_(0,7,2,Nx,Ny)==1) {val=365;}*/
else if(v_d_(3,7,2,Nx,Ny)==1) {val=365;sn=3;}  /* D'=>C */
else if(v_d_(0,12,2,Nx,Ny)==1) {val=366;}  /* G=>H' */
/*else if(v_d_(3,12,2,Nx,Ny)==1) {val=366;sn=3;}*/
/*else if(v_d_(0,12,1,Nx,Ny)==1) {val=367;}*/
else if(v_d_(3,12,1,Nx,Ny)==1) {val=367;sn=3;}  /* H'=>G */
/*else if(v_d_(0,4,0,Nx,Ny)==1) {val=368;}*/
else if(v_d_(3,4,0,Nx,Ny)==1) {val=368;sn=3;}  /* I'=>G */
#endif

#if 1
else if(v_d_(0,1,0,Nx,Ny)==1) {val=369;}  /* B=>C' */
/*else if(v_d_(3,1,0,Nx,Ny)==1) {val=369;sn=3;}*/
/*else if(v_d_(0,5,0,Nx,Ny)==1) {val=370;}*/
else if(v_d_(3,5,0,Nx,Ny)==1) {val=370;sn=3;}  /* C'=>B */
else if(v_d_(0,10,2,Nx,Ny)==1) {val=371;}  /* F=>G' */
/*else if(v_d_(3,10,2,Nx,Ny)==1) {val=371;sn=3;}
/*else if(v_d_(0,12,2,Nx,Ny)==1) {val=372;}*/
else if(v_d_(3,12,2,Nx,Ny)==1) {val=372;sn=3;}  /* G'=>F */
#endif

else
val=399;
}

return val;
}/** check_v **/


void set_vals(int j)
{
int snold;

    enX[j]=X;enY[j]=Y;enX_[j]=X_;enY_[j]=Y_;enSN[j]=sn;

if(0){
snold=sn;
sn=sn_;
```

```
if(pixel[X][Y]==0) getpixel_(X_,Y_,X,Y);
    enX[j]=X;enY[j]=Y;enX_[j]=X_;enY_[j]=Y_;

sn=snold;
}
}/** set_vals **/


int trian_side(int posflag,int task,int Nx,int Ny)
{
int n=RESO,cflag,j,dlt;

/*Nx-=sn*(n-1);*/

if(posflag==200){
if(task==0){
if((c3==ca)||(c2==ca)||(c1==ca)) cflag=1;
else{
Nx-=sn*(n-1);
dlt=Nx-6*(n-1);X_=1*(n-1)+dlt;Y_=1*(n-1);;  /* 101 */
X_+=sn*(n-1);
j=3;rot_h(5,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
j=4;rot_h(5,-1);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=1;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=2;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];

Nx-=sn*(n-1);
dlt=Nx-6*(n-1);X_=1*(n-1)+dlt;Y_=1*(n-1);;  /* 101 */
X_+=sn*(n-1);
j=3;rot_h(5,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
j=4;rot_h(5,-1);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
}
}
else if(posflag==201){
if(task==0){
```

```
if((c4==ca)||(c3==ca)||(c2==ca)) cflag=1;
else{
Nx-=sn*(n-1);
dlt=1*(n-1)-Ny;X_=2*(n-1)+dlt;Y_=3*(n-1);;  /* 102 */
X_+=sn*(n-1);
j=3;rot_h(2,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
j=4;rot_h(2,-1);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=1;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=2;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];

Nx-=sn*(n-1);
dlt=1*(n-1)-Ny;X_=2*(n-1)+dlt;Y_=3*(n-1);;  /* 102 */
X_+=sn*(n-1);
j=3;rot_h(2,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
j=4;rot_h(2,-1);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
}
}
else if(posflag==202){
if(task==0){
if((c1==ca)||(c4==ca)||(c3==ca)) cflag=1;
else{
Nx-=sn*(n-1);
dlt=Nx-6*(n-1);X_=1*(n-1)+dlt;Y_=2*(n-1)+dlt;;  /* 103 */
X_+=sn*(n-1);
j=3;rot_h(3,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
j=4;rot_h(3,-1);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
```

```
j=1;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=2;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];

Nx-=sn*(n-1);
dlt=Nx-6*(n-1);X_=1*(n-1)+dlt;Y_=2*(n-1)+dlt;;  /* 103 */
X_+=sn*(n-1);
j=3;rot_h(3,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
j=4;rot_h(3,-1);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
}
}
else if(posflag==203){
if(task==0){
if((c2==ca)||(c1==ca)||(c4==ca)) cflag=1;
else{
Nx-=sn*(n-1);
dlt=Ny-0*(n-1);X_=1*(n-1);Y_=1*(n-1)+dlt;;  /* 104 */
X_+=sn*(n-1);
j=3;rot_h(4,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
j=4;rot_h(4,-1);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=1;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=2;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];

Nx-=sn*(n-1);
dlt=Ny-0*(n-1);X_=1*(n-1);Y_=1*(n-1)+dlt;;  /* 104 */
X_+=sn*(n-1);
j=3;rot_h(4,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
j=4;rot_h(4,-1);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
}
}


else if(posflag==300){
if(task==0){
```

```
if((c1==ca)||(c4==ca)||(c7==ca)||(c3==ca)) cflag=1;
else{
Nx-=sn*(n-1);
dlt=Nx-1*(n-1);X_=6*(n-1)+dlt;Y_=0*(n-1);;  /* 105 */
X_+=sn*(n-1);
j=4;rot(1,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=1;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=2;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=3;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];

Nx-=sn*(n-1);
dlt=Nx-1*(n-1);X_=6*(n-1)+dlt;Y_=0*(n-1);;  /* 105 */
X_+=sn*(n-1);
j=4;rot(1,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
}
}
else if(posflag==301){
if(task==0){
if((c3==ca)||(c2==ca)||(c5==ca)||(c1==ca)) cflag=1;
else{
Nx-=sn*(n-1);
dlt=3*(n-1)-Nx;X_=7*(n-1);Y_=0*(n-1)+dlt;;  /* 106 */
X_+=sn*(n-1);
j=4;rot(2,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=1;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=2;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=3;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];

Nx-=sn*(n-1);
```

```
dlt=3*(n-1)-Nx;X_=7*(n-1);Y_=0*(n-1)+dlt;;  /* 106 */
X_+=sn*(n-1);
j=4;rot(2,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
}
}
else if(posflag==302){
if(task==0){
if((c7==ca)||(c3==ca)||(c2==ca)||(c5==ca)) cflag=1;
else{
Nx-=sn*(n-1);
dlt=Nx-1*(n-1);X_=6*(n-1)+dlt;Y_=1*(n-1);;  /* 107 */
X_+=sn*(n-1);
j=4;rot(3,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=1;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=2;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=3;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];

Nx-=sn*(n-1);
dlt=Nx-1*(n-1);X_=6*(n-1)+dlt;Y_=1*(n-1);;  /* 107 */
X_+=sn*(n-1);
j=4;rot(3,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
}
}
else if(posflag==303){
if(task==0){
if((c4==ca)||(c7==ca)||(c3==ca)||(c2==ca)) cflag=1;
else{
Nx-=sn*(n-1);
dlt=Ny-1*(n-1);X_=6*(n-1);Y_=0*(n-1)+dlt;;  /* 108 */
X_+=sn*(n-1);
j=4;rot(0,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
```

```
}
else{
j=0;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=1;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=2;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=3;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];

Nx-=sn*(n-1);
dlt=Ny-1*(n-1);X_=6*(n-1);Y_=0*(n-1)+dlt;;  /* 108 */
X_+=sn*(n-1);
j=4;rot(0,0);X=X_+tmp0;Y=Y_+tmp1;
set_vals(j);
}
}


if(task==0) return cflag;
else        return j;
}/** trian_side **/


int trian_vtx(int posflag,int task,int Nx,int Ny)
{
int n=RESO,cflag,j,dsn;

/* inside vtx(square) */
if(posflag==250){  /* s-0 */
if(task==0){
if((c2==ca)||(c1==ca)) cflag=1;
else{
X_=1*(n-1);Y_=1*(n-1);;
X_+=sn*(n-1);
j=2;X=X_+0;Y=Y_-1;
set_vals(j);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_-1;Y=Y_+0;
set_vals(j);

cflag=0;
for(j=2;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=1;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
```

```
X_=1*(n-1);Y_=1*(n-1);;
X_+=sn*(n-1);
j=2;X=X_+0;Y=Y_-1;
set_vals(j);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_-1;Y=Y_+0;
set_vals(j);
}
}
else if(posflag==251){  /* s-1 */
if(task==0){
if((c3==ca)||(c2==ca)) cflag=1;
else{
X_=3*(n-1);Y_=3*(n-1);;
X_+=sn*(n-1);
j=2;X=X_+0;Y=Y_+1;
set_vals(j);
j=3;X=X_+1;Y=Y_+1;
set_vals(j);

X_=2*(n-1);Y_=1*(n-1);;
X_+=sn*(n-1);
j=4;X=X_-1;Y=Y_-1;
set_vals(j);

cflag=0;
for(j=2;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=1;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];

X_=3*(n-1);Y_=3*(n-1);;
X_+=sn*(n-1);
j=2;X=X_+0;Y=Y_+1;
set_vals(j);
j=3;X=X_+1;Y=Y_+1;
set_vals(j);

X_=2*(n-1);Y_=1*(n-1);;
X_+=sn*(n-1);
j=4;X=X_-1;Y=Y_-1;
set_vals(j);
```

```
}
}
else if(posflag==252){  /* s-2 */
if(task==0){
if((c4==ca)||(c3==ca)) cflag=1;
else{
X_=2*(n-1);Y_=3*(n-1);;
X_+=sn*(n-1);
j=2;X=X_-1;Y=Y_-0;
set_vals(j);
j=3;X=X_-0;Y=Y_+1;
set_vals(j);
j=4;X=X_+1;Y=Y_+1;
set_vals(j);

cflag=0;
for(j=2;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=1;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];

X_=2*(n-1);Y_=3*(n-1);;
X_+=sn*(n-1);
j=2;X=X_-1;Y=Y_-0;
set_vals(j);
j=3;X=X_-0;Y=Y_+1;
set_vals(j);
j=4;X=X_+1;Y=Y_+1;
set_vals(j);
}
}
else if(posflag==253){  /* s-3 */
if(task==0){
if((c1==ca)||(c4==ca)) cflag=1;
else{
X_=1*(n-1);Y_=2*(n-1);;
X_+=sn*(n-1);
j=2;X=X_-1;Y=Y_-1;
set_vals(j);
j=3;X=X_-1;Y=Y_+0;
set_vals(j);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);
```

```
cflag=0;
for(j=2;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=1;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];

X_=1*(n-1);Y_=2*(n-1);;
X_+=sn*(n-1);
j=2;X=X_-1;Y=Y_-1;
set_vals(j);
j=3;X=X_-1;Y=Y_+0;
set_vals(j);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);
}
}


/* posflag>=350 */
/* inside vtx(delta) */
else if(posflag==350){  /* d3-2 */
if(task==0){
if((c1==ca)||(c4==ca)||(c7==ca)||(c3==ca)||(c2==ca)) cflag=1;
else{
cflag=0;
}
}
else{
j=0;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=1;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=2;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=3;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=4;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
}
}
else if(posflag==351){  /* d4-1 */
if(task==0){
if((c4==ca)||(c7==ca)||(c3==ca)) cflag=1;
else{
X_=3*(n-1);Y_=3*(n-1);;
X_+=sn*(n-1);
j=3;X=X_-1;Y=Y_-0;
set_vals(j);
j=4;X=X_-0;Y=Y_+1;
```

```
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=1;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=2;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];

X_=3*(n-1);Y_=3*(n-1);;
X_+=sn*(n-1);
j=3;X=X_-1;Y=Y_-0;
set_vals(j);
j=4;X=X_-0;Y=Y_+1;
set_vals(j);
}
}
else if(posflag==352){  /* d11-0 */
if(task==0){
if((c3==ca)||(c2==ca)||(c5==ca)) cflag=1;
else{
X_=2*(n-1);Y_=1*(n-1);;
X_+=sn*(n-1);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_-1;Y=Y_+0;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=1;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=2;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];

X_=2*(n-1);Y_=1*(n-1);;
X_+=sn*(n-1);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_-1;Y=Y_+0;
set_vals(j);
}
```

```
}
else if(posflag==353){  /* d8-1 */
if(task==0){
if((c7==ca)||(c3==ca)||(c2==ca)||(c5==ca)||(c1==ca)) cflag=1;
else{
cflag=0;
}
}
else{
j=0;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=1;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=2;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=3;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=4;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
}
}
else if(posflag==354){  /* d6-1 */
if(task==0){
if((c4==ca)||(c7==ca)||(c3==ca)||(c2==ca)||(c5==ca)) cflag=1;
else{
cflag=0;
}
}
else{
j=0;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=1;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=2;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=3;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=4;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
}
}


/* outside vtx */
else if(posflag==355){  /* A=>B' */
if(task==0){
if((c3==ca)||(c2==ca)||(c5==ca)||(c1==ca)) cflag=1;
else{
X_=0*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+1;Y=Y_+1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=1;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=2;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=3;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];

X_=0*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+1;Y=Y_+1;
set_vals(j);
}
}
else if(posflag==356){  /* B'=>A */
if(task==0){
if((c2==ca)||(c5==ca)||(c1==ca)) cflag=1;
else{
X_=1*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+1;Y=Y_+1;
set_vals(j);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=1;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=2;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];

X_=1*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+1;Y=Y_+1;
set_vals(j);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);
}
}
else if(posflag==357){  /* E=>F' */
```

```
if(task==0){
if((c5==ca)||(c1==ca)||(c4==ca)||(c7==ca)) cflag=1;
else{
X_=2*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+0;Y=Y_-1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=1;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=2;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=3;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];

X_=2*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+0;Y=Y_-1;
set_vals(j);
}
}
else if(posflag==358){  /* F'=>E */
if(task==0){
if((c1==ca)||(c4==ca)||(c7==ca)) cflag=1;
else{
X_=1*(n-1);Y_=3*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+0;Y=Y_-1;
set_vals(j);
j=4;X=X_+1;Y=Y_+0;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=1;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=2;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
```

```
X_=1*(n-1);Y_=3*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+0;Y=Y_-1;
set_vals(j);
j=4;X=X_+1;Y=Y_+0;
set_vals(j);
}
}


else if(posflag==359){  /* D=>E' */
if(task==0){
if((c5==ca)||(c1==ca)||(c4==ca)) cflag=1;
else{
X_=1*(n-1);Y_=3*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+0;Y=Y_-1;
set_vals(j);
j=4;X=X_+1;Y=Y_+0;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=1;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=2;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];

X_=1*(n-1);Y_=3*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+0;Y=Y_-1;
set_vals(j);
j=4;X=X_+1;Y=Y_+0;
set_vals(j);
}
}
else if(posflag==360){  /* E'=>D */
if(task==0){
if((c5==ca)||(c1==ca)||(c4==ca)||(c7==ca)) cflag=1;
else{
```

```
X_=0*(n-1);Y_=2*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+1;Y=Y_+0;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=1;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=2;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=3;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];

X_=0*(n-1);Y_=2*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+1;Y=Y_+0;
set_vals(j);
}
}
else if(posflag==361){  /* H(0)=>A'(3) */
if(task==0){
if((c7==ca)||(c3==ca)) cflag=1;
else{
X_=2*(n-1);Y_=0*(n-1);;
dsn=0;
X_+=dsn*(n-1);
j=2;X=X_-1;Y=Y_+0;
set_vals(j);

X_=1*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+1;Y=Y_+1;
set_vals(j);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);

cflag=0;
for(j=2;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
```

```
j=1;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=2;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];

X_=2*(n-1);Y_=0*(n-1);;
j=/*2*/0;X=X_-1;Y=Y_+0;
set_vals(j);

X_=1*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+1;Y=Y_+1;
set_vals(j);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);
}
}
else if(posflag==362){  /* I(0)=>A'(3) */
if(task==0){
if((c3==ca)||(c2==ca)) cflag=1;
else{
X_=4*(n-1);Y_=4*(n-1);;
dsn=0;
X_+=dsn*(n-1);
j=2;X=X_-1;Y=Y_+0;
set_vals(j);

X_=1*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+1;Y=Y_+1;
set_vals(j);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);

cflag=0;
for(j=2;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=1;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];

X_=4*(n-1);Y_=4*(n-1);;
j=2;X=X_-1;Y=Y_+0;
set_vals(j);
```

```
X_=1*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+1;Y=Y_+1;
set_vals(j);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);
}
}
else if(posflag==363){  /* A'=>I */
if(task==0){
if((c3==ca)||(c2==ca)||(c5==ca)||(c1==ca)) cflag=1;
else{
X_=2*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=1;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=2;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=3;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];

X_=2*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+0;Y=Y_+1;
set_vals(j);
}
}


else if(posflag==364){  /* C=>D' */
if(task==0){
if((c2==ca)||(c5==ca)||(c1==ca)||(c4==ca)) cflag=1;
else{
X_=0*(n-1);Y_=2*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+1;Y=Y_+0;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=1;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=2;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=3;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];

X_=0*(n-1);Y_=2*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+1;Y=Y_+0;
set_vals(j);
}
}
else if(posflag==365){  /* D'=>C */
if(task==0){
if((c5==ca)||(c1==ca)||(c4==ca)) cflag=1;
else{
X_=0*(n-1);Y_=1*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+1;Y=Y_-0;
set_vals(j);
j=4;X=X_+1;Y=Y_+1;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=1;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=2;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];

X_=0*(n-1);Y_=1*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+1;Y=Y_-0;
set_vals(j);
j=4;X=X_+1;Y=Y_+1;
```

```
set_vals(j);
}
}
else if(posflag==366){  /* G=>H' */
if(task==0){
if((c1==ca)||(c4==ca)||(c7==ca)||(c3==ca)) cflag=1;
else{
X_=4*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_-1;Y=Y_-1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=1;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=2;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=3;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];

X_=4*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_-1;Y=Y_-1;
set_vals(j);
}
}
else if(posflag==367){  /* H'(3)=>G(0) */
if(task==0){
if((c7==ca)||(c3==ca)) cflag=1;
else{
X_=2*(n-1);Y_=0*(n-1);;
dsn=3;
X_+=dsn*(n-1);
j=2;X=X_-1;Y=Y_+0;
set_vals(j);

X_=3*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_+0;Y=Y_-1;
set_vals(j);

cflag=0;
for(j=2;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=1;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=2;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];

X_=2*(n-1);Y_=0*(n-1);;
dsn=3;
X_+=dsn*(n-1);
j=/*2*/0;X=X_-1;Y=Y_+0;
set_vals(j);

X_=3*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_+0;Y=Y_-1;
set_vals(j);
}
}
else if(posflag==368){  /* I'(3)=>G(0) */
if(task==0){
if((c3==ca)||(c2==ca)) cflag=1;
else{
X_=4*(n-1);Y_=4*(n-1);;
dsn=3;
X_+=dsn*(n-1);
j=2;X=X_-1;Y=Y_+0;
set_vals(j);

X_=3*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_+0;Y=Y_-1;
set_vals(j);

cflag=0;
for(j=2;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
```

```
}
else{
j=0;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=1;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];

X_=4*(n-1);Y_=4*(n-1);;
dsn=3;
X_+=dsn*(n-1);
j=2;X=X_-1;Y=Y_+0;
set_vals(j);

X_=3*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_+0;Y=Y_-1;
set_vals(j);
}
}



else if(posflag==369){  /* B=>C' */
if(task==0){
if((c2==ca)||(c5==ca)||(c1==ca)) cflag=1;
else{
X_=0*(n-1);Y_=1*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_+1;Y=Y_+0;
set_vals(j);
j=4;X=X_+1;Y=Y_+1;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=1;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=2;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];

X_=0*(n-1);Y_=1*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
```

```
j=3;X=X_+1;Y=Y_+0;
set_vals(j);
j=4;X=X_+1;Y=Y_+1;
set_vals(j);
}
}
else if(posflag==370){  /* C'=>B */
if(task==0){
if((c2==ca)||(c5==ca)||(c1==ca)||(c4==ca)) cflag=1;
else{
X_=0*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+1;Y=Y_+1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=1;enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=2;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=3;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];

X_=0*(n-1);Y_=0*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_+1;Y=Y_+1;
set_vals(j);
}
}
else if(posflag==371){  /* F=>G' */
if(task==0){
if((c1==ca)||(c4==ca)||(c7==ca)) cflag=1;
else{
X_=3*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_+0;Y=Y_-1;
set_vals(j);

cflag=0;
for(j=3;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=1;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=2;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];

X_=3*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=3;X=X_-1;Y=Y_-1;
set_vals(j);
j=4;X=X_+0;Y=Y_-1;
set_vals(j);
}
}
else if(posflag==372){  /* G'=>F */
if(task==0){
if((c1==ca)||(c4==ca)||(c7==ca)||(c3==ca)) cflag=1;
else{
X_=2*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_-0;Y=Y_-1;
set_vals(j);

cflag=0;
for(j=4;j<=4;j++) {if(rpixel(enX[j],enY[j])==ca) {cflag=1;break;}}
}
}
else{
j=0;enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=1;enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=2;enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=3;enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];

X_=2*(n-1);Y_=4*(n-1);;
if(sn==0) dsn=3;else dsn=0;
X_+=dsn*(n-1);
j=4;X=X_-0;Y=Y_-1;
set_vals(j);
}
}
```

```
if(task==0) return cflag;
else        return j;
}/** trian_vtx **/


char rpixel(int i,int j)
{
if(c_trans==0) return pixel[i][j];
else{
if(!zeroFill)   return pixel_[i][j];
else            return pixel[i][j];
}
}/** rpixel **/


void wpixel(int i,int j,char val_w)
{
if(c_trans==0) pixel[i][j]=val_w;
else{
if(!zeroFill)   pixel_[i][j]=val_w;
else            pixel[i][j]=val_w;
}
}/** wpixel **/


void set_DP(int x,int y,int n_)
{
int dx;

if((n_-1)%3!=0) return;

dx=(n_-1)/3;
wpixel(x*(RESO-1)+dx,y*(RESO-1)+dx*2,-1);
id[x*(RESO-1)+dx][y*(RESO-1)+dx*2]=-1;
}/** set_DP **/


int cag_r(int nax_,int nay_,int color_)
{
int i,j,dx,dy,n=RESO,jmax,sn;
int flag_[CPMAX],flag_pp[CPMAX],acolor[8];
int nx[CPMAX],ny[CPMAX],nx_[CPMAX],ny_[CPMAX],nax[8],nay[8];
int cp,ssize,posflag,pos,count,cflag,algo;
int nxp,nxm,nyp,nym,Nx,Ny;
int sntmp1,sntmp2;
long val;
```

```
ssize=sizeof(ss);
if(c_trans==0) cp=CPMAX;else cp=1;

fieldflag=1;
if(0){
/* outside vtx:8 */
sn=0;
i=sn+0;j=0;pp(i,j,8);
i=sn+1;j=0;pp(i,j,8);
i=sn+2;j=0;pp(i,j,8);
i=sn+0;j=1;pp(i,j,8);
/*i=sn+1;j=1;pp(i,j,8);*/
/*i=sn+2;j=1;pp(i,j,8);*/
i=sn+0;j=2;pp(i,j,8);
/*i=sn+1;j=2;pp(i,j,8);*/
i=sn+1;j=3;pp(i,j,8);
/*i=sn+2;j=3;pp(i,j,8);*/
/*i=sn+3;j=3;pp(i,j,8);*/
i=sn+2;j=4;pp(i,j,8);
i=sn+3;j=4;pp(i,j,8);
i=sn+4;j=4;pp(i,j,8);

/*i=sn+6;j=0;pp(i,j,8);
i=sn+7;j=0;pp(i,j,8);
i=sn+7;j=1;pp(i,j,8);
i=sn+6;j=1;pp(i,j,8);*/

sn=3;
i=sn+0;j=0;pp(i,j,8);
i=sn+1;j=0;pp(i,j,8);
i=sn+2;j=0;pp(i,j,8);
i=sn+0;j=1;pp(i,j,8);
/*i=sn+1;j=1;pp(i,j,8);*/
/*i=sn+2;j=1;pp(i,j,8);*/
i=sn+0;j=2;pp(i,j,8);
/*i=sn+1;j=2;pp(i,j,8);*/
i=sn+1;j=3;pp(i,j,8);
/*i=sn+2;j=3;pp(i,j,8);*/
/*i=sn+3;j=3;pp(i,j,8);*/
i=sn+2;j=4;pp(i,j,8);
i=sn+3;j=4;pp(i,j,8);
i=sn+4;j=4;pp(i,j,8);

/*i=sn+6;j=0;pp(i,j,8);
i=sn+7;j=0;pp(i,j,8);
```

```
i=sn+7;j=1;pp(i,j,8);
i=sn+6;j=1;pp(i,j,8);*/
}
else if(1){
sn=0;
/* 1st pair:A-E */
if(0){
i=sn+1;j=0;pp(i,j,8);
i=sn+1;j=3;pp(i,j,8);
}

/* 4th pair:D-H(I) */
if(0){
i=sn+0;j=2;pp(i,j,8);
}
if(0){
i=sn+4;j=4;pp(i,j,8);
i=sn+2;j=0;pp(i,j,8);
}

/* 3rd pair:C-G */
if(0){
i=sn+0;j=1;pp(i,j,8);
i=sn+3;j=4;pp(i,j,8);
}

/* 2nd pair:B-F */
if(0){
i=sn+0;j=0;pp(i,j,8);
i=sn+2;j=4;pp(i,j,8);
}

/*i=sn+1;j=1;pp(i,j,8);*/
/*i=sn+2;j=1;pp(i,j,8);*/
/*i=sn+1;j=2;pp(i,j,8);*/
/*i=sn+2;j=3;pp(i,j,8);*/
/*i=sn+3;j=3;pp(i,j,8);*/

/*i=sn+6;j=0;pp(i,j,8);
i=sn+7;j=0;pp(i,j,8);
i=sn+7;j=1;pp(i,j,8);
i=sn+6;j=1;pp(i,j,8);*/

sn=3;
/* 1st pair:A-E */
if(0){
```

```
i=sn+0;j=0;pp(i,j,8);
i=sn+2;j=4;pp(i,j,8);
}

/* 4th pair:D-H(I) */
if(0){
i=sn+1;j=3;pp(i,j,8);
}
if(0){
i=sn+1;j=0;pp(i,j,8);
}

/* 3rd pair:C-G */
if(0){
i=sn+0;j=2;pp(i,j,8);
i=sn+4;j=4;pp(i,j,8);
i=sn+2;j=0;pp(i,j,8);
}

/* 2nd pair:B-F */
if(0){
i=sn+0;j=1;pp(i,j,8);
i=sn+3;j=4;pp(i,j,8);
}

/*i=sn+1;j=1;pp(i,j,8);*/
/*i=sn+2;j=1;pp(i,j,8);*/
/*i=sn+1;j=2;pp(i,j,8);*/
/*i=sn+2;j=3;pp(i,j,8);*/
/*i=sn+3;j=3;pp(i,j,8);*/

/*i=sn+6;j=0;pp(i,j,8);
i=sn+7;j=0;pp(i,j,8);
i=sn+7;j=1;pp(i,j,8);
i=sn+6;j=1;pp(i,j,8);*/
}
fieldflag=0;

for(i=0;i<CPMAX;i++){
rcount[i]=0;
flag_[i]=1;
fp_mem[i]=0;
}

/*999*/
if(!zeroFill) ca=/*15*/16;
```

```
else          ca=zg;
sn=0;

/* only in square */
if(CPMAX==4 && DIV==1){
}
else if(DIV==0){
dx=0*(n-1);dy=0;
nax[0]=6*(n-1)+1+dx+DSP      ;nay[0]=0*(n-1)+1+dy+DSP;
nax[1]=7*(n-1)-1+dx-DSP      ;nay[1]=0*(n-1)+1+dy-DSP;
nax[2]=7*(n-1)-1+dx-DSP      ;nay[2]=1*(n-1)-1+dy+DSP;
nax[3]=6*(n-1)+1+dx-DSP      ;nay[3]=1*(n-1)-1+dy-DSP;
}
else if(DIV==1){
/* J'K'L'M' */
dx=0;dy=0;
nax[0]=6*(n-1)+1+dx+DSP      ;nay[0]=0*(n-1)+1+dy+DSP;
nax[1]=7*(n-1)-1+dx-DSP      ;nay[1]=0*(n-1)+1+dy-DSP;
nax[2]=7*(n-1)-1+dx-DSP      ;nay[2]=1*(n-1)-1+dy+DSP;
nax[3]=6*(n-1)+1+dx-DSP      ;nay[3]=1*(n-1)-1+dy-DSP;
/* O'P'Q'R' */
dx=3*(n-1);dy=0;
nax[4]=6*(n-1)+1+dx+DSP      ;nay[4]=0*(n-1)+1+dy+DSP;
nax[5]=7*(n-1)-1+dx-DSP      ;nay[5]=0*(n-1)+1+dy-DSP;
nax[6]=7*(n-1)-1+dx-DSP      ;nay[6]=1*(n-1)-1+dy+DSP;
nax[7]=6*(n-1)+1+dx-DSP      ;nay[7]=1*(n-1)-1+dy-DSP;
}

acolor[0]=9;acolor[1]=10;acolor[2]=11;acolor[3]=12;
acolor[4]=13;acolor[5]=14;acolor[6]=7;acolor[7]=8;


if(c_trans>0){
nax[0]=nax_;nay[0]=nay_;
acolor[0]=color_;
}


i=0;
while(1){
if(flag_[i]){                       /* CP_? */
/*if(i<=2) sn=sn1;else sn=sn2;*/
ig=i;

nx[i]=nax[i];ny[i]=nay[i];
/*sn=0;*/
```

```
putpixel_(nx[i],ny[i],acolor[i]);
if(c_trans==2){
searchflag=1;
search_i(nx[0],ny[0],acolor[0]);
searchflag=0;
}
}/**if(flag_[i])**/

i++;
if(c_trans==0) {if(i==CPMAX) break;}
else break;
}/**while(1)**/

i=0;
while(1){
if(flag_[i]){                       /* CP_? */
nx_[i]=nax[i];ny_[i]=nay[i];
/*if(i<=2) sn=sn1;else sn=sn2;*/
ig=i;

if(c_trans==0){
/* square only */
     if(i%4==0) {nax[i]++;}
else if(i%4==1) {nay[i]++;}
else if(i%4==2) {nax[i]--;}
else if(i%4==3) {nay[i]--;}
}
else{
nx[i]=nax[i];ny[i]=nay[i];
posflag=check_v(nx[i],ny[i]);
nxp=nx[i]+1;nyp=ny[i]+1;nxm=nx[i]-1;nym=ny[i]-1;

if(posflag<300){
c1=getpixel_(nx[i],ny[i],nxp,ny[i]);
x[1]=X;y[1]=Y;x_[1]=X_;y_[1]=Y_;jmp[1]=sn_;
c2=getpixel_(nx[i],ny[i],nx[i],nyp);
x[2]=X;y[2]=Y;x_[2]=X_;y_[2]=Y_;jmp[2]=sn_;
c3=getpixel_(nx[i],ny[i],nxm,ny[i]);
x[3]=X;y[3]=Y;x_[3]=X_;y_[3]=Y_;jmp[3]=sn_;
c4=getpixel_(nx[i],ny[i],nx[i],nym);
x[4]=X;y[4]=Y;x_[4]=X_;y_[4]=Y_;jmp[4]=sn_;

if(c1==ca) nax[0]++;
else if(c2==ca) nay[0]++;
else if(c3==ca) nax[0]--;
else if(c4==ca) nay[0]--;
}
else{
c1=getpixel_(nx[i],ny[i],nxp,ny[i]);
x[1]=X;y[1]=Y;x_[1]=X_;y_[1]=Y_;jmp[1]=sn_;
c5=getpixel_(nx[i],ny[i],nxp,nyp);
x[5]=X;y[5]=Y;x_[5]=X_;y_[5]=Y_;jmp[5]=sn_;
c2=getpixel_(nx[i],ny[i],nx[i],nyp);
x[2]=X;y[2]=Y;x_[2]=X_;y_[2]=Y_;jmp[2]=sn_;
c3=getpixel_(nx[i],ny[i],nxm,ny[i]);
x[3]=X;y[3]=Y;x_[3]=X_;y_[3]=Y_;jmp[3]=sn_;
c7=getpixel_(nx[i],ny[i],nxm,nym);
x[7]=X;y[7]=Y;x_[7]=X_;y_[7]=Y_;jmp[7]=sn_;
c4=getpixel_(nx[i],ny[i],nx[i],nym);
x[4]=X;y[4]=Y;x_[4]=X_;y_[4]=Y_;jmp[4]=sn_;

if(c1==ca) nax[0]++;
else if(c2==ca) nay[0]++;
else if(c3==ca) nax[0]--;
else if(c4==ca) nay[0]--;
else if(c5==ca) {nax[0]++;nay[0]++;}
else if(c7==ca) {nax[0]--;nay[0]--;}
}
}

nx[i]=nax[i];ny[i]=nay[i];

/*sn=0;*/
putpixel_(nx[i],ny[i],acolor[i]);
if(c_trans==2){
searchflag=1;
search_i(nx[0],ny[0],acolor[0]);
searchflag=0;
}
}/**if(flag_[i])**/

i++;
if(c_trans==0) {if(i==CPMAX) break;}
else break;
}/**while(1)**/

if(c_trans>0){
i=0;
s.xx=nx_[i];s.yy=ny_[i];s.xx_=nx[i];s.yy_=ny[i];s.sn=sn;fwrite_mem(i);
}

if(c_trans==0 && GRPH==1 && cnt==0) use_subroop();
```

```
cnt++;
/***************************** while(cp) -> ******************************/

while(cp){
kbhit_();
if(refill==0) break;
if(0){
use_subroop();
/*printf(" \n");*/
}

algo=random_(2);

i=0;
while(1){

if(flag_[i]){                      /* CP_? */
/*sn=0;*/
ig=i;

Nx=nx[i];Ny=ny[i];
posflag=check_v(Nx,Ny);
nxp=nx[i]+1;nyp=ny[i]+1;nxm=nx[i]-1;nym=ny[i]-1;

if(posflag<300){
c1=getpixel_(nx[i],ny[i],nxp,ny[i]);
x[1]=X;y[1]=Y;x_[1]=X_;y_[1]=Y_;jmp[1]=sn_;
c2=getpixel_(nx[i],ny[i],nx[i],nyp);
x[2]=X;y[2]=Y;x_[2]=X_;y_[2]=Y_;jmp[2]=sn_;
c3=getpixel_(nx[i],ny[i],nxm,ny[i]);
x[3]=X;y[3]=Y;x_[3]=X_;y_[3]=Y_;jmp[3]=sn_;
c4=getpixel_(nx[i],ny[i],nx[i],nym);
x[4]=X;y[4]=Y;x_[4]=X_;y_[4]=Y_;jmp[4]=sn_;
}
else{
c1=getpixel_(nx[i],ny[i],nxp,ny[i]);
x[1]=X;y[1]=Y;x_[1]=X_;y_[1]=Y_;jmp[1]=sn_;
c5=getpixel_(nx[i],ny[i],nxp,nyp);
x[5]=X;y[5]=Y;x_[5]=X_;y_[5]=Y_;jmp[5]=sn_;
c2=getpixel_(nx[i],ny[i],nx[i],nyp);
x[2]=X;y[2]=Y;x_[2]=X_;y_[2]=Y_;jmp[2]=sn_;
c3=getpixel_(nx[i],ny[i],nxm,ny[i]);
x[3]=X;y[3]=Y;x_[3]=X_;y_[3]=Y_;jmp[3]=sn_;
c7=getpixel_(nx[i],ny[i],nxm,nym);
x[7]=X;y[7]=Y;x_[7]=X_;y_[7]=Y_;jmp[7]=sn_;
c4=getpixel_(nx[i],ny[i],nx[i],nym);
```

```
x[4]=X;y[4]=Y;x_[4]=X_;y_[4]=Y_;jmp[4]=sn_;
}



if(posflag==299){
if((c1==ca)||(c2==ca)||(c3==ca)||(c4==ca)) cflag=1;
else cflag=0;
}
else if(posflag==399){
if((c1==ca)||(c2==ca)||(c3==ca)||(c4==ca)||(c5==ca)||(c7==ca)) cflag=1;
else cflag=0;
}
else if((posflag>=200 && posflag<250) || (posflag>=300 && posflag<350))
cflag=trian_side(posflag,0,Nx,Ny);
else
cflag=trian_vtx(posflag,0,Nx,Ny);



if(cflag){
s.xx=nx[i];s.yy=ny[i];s.xx_=nx_[i];s.yy_=ny_[i];s.sn=sn;fwrite_mem(i);

if(posflag==299){
j=-1;

if(c1>=0){
j++;  /* ca1 */
    enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
}
if(c4>=0){
j++;  /* ca4 */
    enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
}
if(c3>=0){
j++;  /* ca3 */
    enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
}
if(c2>=0){
j++;  /* ca2 */
    enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
}
}
else if(posflag==399){
j=-1;
```

```
if(c1>=0){
j++;  /* ca1 */
    enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
}
if(c4>=0){
j++;  /* ca4 */
    enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
}
if(c7>=0){
j++;  /* ca7 */
    enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
}
if(c3>=0){
j++;  /* ca3 */
    enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
}
if(c2>=0){
j++;  /* ca2 */
    enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
}
if(c5>=0){
j++;  /* ca5 */
    enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
}
}
else if((posflag>=200 && posflag<250) || (posflag>=300 && posflag<350))
j=trian_side(posflag,1,Nx,Ny);
else
j=trian_vtx(posflag,1,Nx,Ny);

jmax=j;

for(j=0;j<=jmax;j++){
if(enX[j]==nx_[i] && enY[j]==ny_[i]) {pos=j;break;}
}
/*printf(" jmax:%d\n",jmax);
if(jmax==4){
for(j=0;j<=jmax;j++)
printf(" j:%d; %d %d; %d; %d %d\n",
           j,enX[j],enY[j],pixel[enX[j]][enY[j]],enX_[j],enY_[j]);
printf(" pos:%d %d %d\n",pos,enX[pos],enY[pos]);
}*/

if(algo==0 || YOUR_ART==1){
/* CW */
count=0;
```

```
for(j=pos;;){
if(rpixel(fen("X",j,jmax),fen("Y",j,jmax))!=ca &&
   rpixel(fen("X",j-1,jmax),fen("Y",j-1,jmax))==ca){
nx[i]=fen("X",j-1,jmax);ny[i]=fen("Y",j-1,jmax);
nx_[i]=fen("X_",j-1,jmax);ny_[i]=fen("Y_",j-1,jmax);
if(i<=CPHALF-1) sntmp1=fen("SN",j-1,jmax);
else sntmp2=fen("SN",j-1,jmax);
break;
}

j--;if(j<0) j=jmax;
count++;if(count==jmax+1) {printf(" ?CW\n");break;}
}/**for()**/
}
else{
/* CCW */
count=0;
for(j=pos;;){
if(rpixel(fen("X",j,jmax),fen("Y",j,jmax))!=ca &&
   rpixel(fen("X",j+1,jmax),fen("Y",j+1,jmax))==ca){
nx[i]=fen("X",j+1,jmax);ny[i]=fen("Y",j+1,jmax);
nx_[i]=fen("X_",j+1,jmax);ny_[i]=fen("Y_",j+1,jmax);
if(i<=CPHALF-1) sntmp1=fen("SN",j+1,jmax);
else sntmp2=fen("SN",j+1,jmax);
break;
}

j++;if(j>jmax) j=0;
count++;if(count==jmax+1) {printf(" ?CCW\n");break;}
}/**for()**/
}

if(1){
/*if(i<2){
sn=sntmp1;
}
else if(i==2){
sn=sntmp1;
sn1=sntmp1;
}
else if(i>2 && i<5){
sn=sntmp2;
}
else if(i==5){
sn=sntmp2;
sn2=sntmp2;
}*/

putpixel_(nx[i],ny[i],acolor[i]);
if(c_trans==2){
searchflag=1;
search_i(nx[0],ny[0],acolor[0]);
searchflag=0;
}
}

flag_pp[i]=1;
}/**if(cflag)**/
else{
val=ftell_mem(i);
/*printf(" %ld\n",val);*/
if(val==0) {flag_[i]=0;cp--;if(cp==0) break;}
fread_mem(i);
nx[i]=s.xx;ny[i]=s.yy;nx_[i]=s.xx_;ny_[i]=s.yy_;
/*if(i<=2) sn1=s.sn;
else sn2=s.sn;*/
flag_pp[i]=0;
}/**else(c1,c2,c3,c4)**/
}/**if(flag_[i])**/

i++;
if(c_trans==0) {if(i==CPMAX) break;}
else break;
}/**while(1)**/
}/**while(cp)**/

if(c_trans==1){
if(rcount[0]>=RCMAX/**CPMAX-bdrnum*/) return 1;
else return 0;
}
else return 0;
}/** cag_r **/
```