

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. Icosahedron

Figure 1 is a development of icosahedron. The number is the number of the vertex and the following are combinations of numbers of vertexes, two sides in correspondence.

- 131, 132, 133, 134, 135 ; 141, 142, 143, 144, 145 ; 1, 6 ; 7, 12 ;
- 2-131, 2-132 ; 3-132, 3-133 ; 4-133, 4-134 ; 5-134, 5-135 ; 1-131, 6-135;8-141, 8-142 ;
- 9-142, 9-143 ; 10-143, 10-144 ; 11-144, 11-145;7-141, 12-145;1-7, 6-12

The correspondence between two sides of which end points are vertex 2 ~ 5, 8 ~ 11 is the same as cut in the last and the correspondence between two sides is parallel movement in 1-7, 6-12.

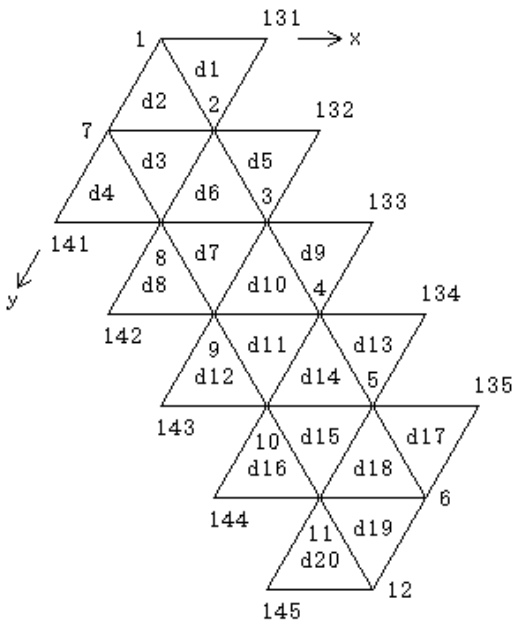
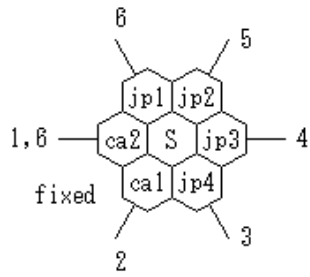


Figure 1

2. Neighborhood view

Figure 2 is neighborhood view on vertex 131. In vertex 131 ~ 135, the five regular rectangles which include them are combined.

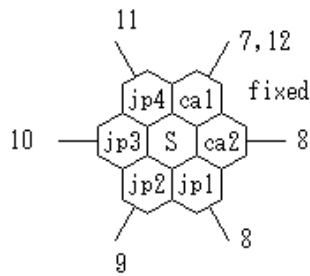


$S: (n-1, 0)$      $S': (X', Y')$   
 $X' = 5(n-1)$      $Y' = 4(n-1)$   
 $jp1: (X', Y'+1)$   
 $X' = 4(n-1)$      $Y' = 3(n-1)$   
 $jp2: (X', Y'+1)$   
 $X' = 3(n-1)$      $Y' = 2(n-1)$   
 $jp3: (X', Y'+1)$   
 $X' = 2(n-1)$      $Y' = n-1$   
 $jp4: (X', Y'+1)$

vertex 131

Figure 2

Figure 3 is neighborhood view on vertex 141. In vertex 141 ~ 145, the five regular rectangles which include them are combined.

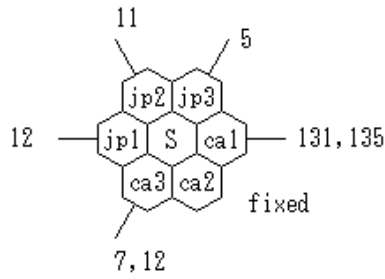


$S: (0, 2(n-1))$      $S': (X', Y')$   
 $X' = n-1$      $Y' = 3(n-1)$   
 $jp1: (X', Y'-1)$   
 $X' = 2(n-1)$      $Y' = 4(n-1)$   
 $jp2: (X', Y'-1)$   
 $X' = 3(n-1)$      $Y' = 5(n-1)$   
 $jp3: (X', Y'-1)$   
 $X' = 4(n-1)$      $Y' = 6(n-1)$   
 $jp4: (X', Y'-1)$

vertex 141

Figure 3

Figure 4 is neighborhood view on vertex 1. In vertex 1, the two regular rectangles which are around vertex 1 and the three regular rectangles which are around vertex 6 are combined and in vertex 6, the three regular rectangles which are around vertex 6 and the two regular rectangles which are around vertex 1 are combined.

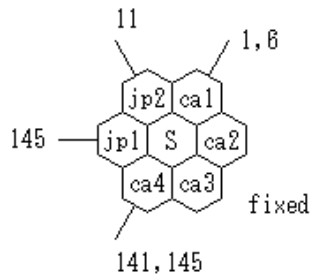


$S:(0,0)$      $S':(X',Y')$   
 $X'=5(n-1)$      $Y'=5(n-1)$   
 $jp1:(X',Y'+1)$   
 $jp2:(X'-1,Y')$   
 $jp3:(X'-1,Y'-1)$

vertex 1

Figure 4

Figure 5 is neighborhood view on vertex 7. In vertex 7, the three regular rectangles which are around vertex 7 and the two regular rectangles which are around vertex 12 are combined and in vertex 12, the two regular rectangles which are around vertex 12 and the three regular rectangles which are around vertex 7 are combined.

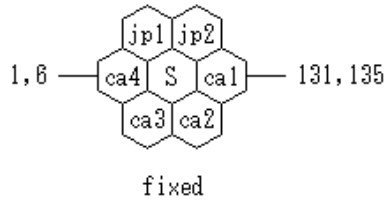


$S:(0,n-1)$      $S':(X',Y')$   
 $X'=5(n-1)$      $Y'=6(n-1)$   
 $jp1:(X'-1,Y')$   
 $jp2:(X'-1,Y'-1)$

vertex 7

Figure 5

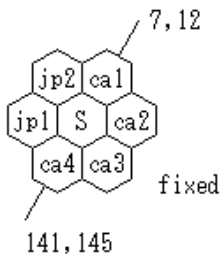
Figure 6 is neighborhood view on side 1-131. The two end points of the side are not included among pixels of the side. In side 1-131, the regular triangle which includes side 6-135 is combined and in side 6-135, the regular triangle which includes side 1-131 is combined. In a side of which end point is vertex 2 ~ 5, the regular triangle which includes the side of which end point is the same as the vertex is combined.



$S:(x,y)$      $S':(X',Y')$   
 $X'=5(n-1)$      $Y'=y+5(n-1)-x$   
 $jp1:(X'-1,Y')$   
 $jp2:(X'-1,Y'-1)$   
 side 1-131

Figure 6

Figure 7 is neighborhood view on side 7-141. In side 7-141, the regular triangle which includes side 12-145 is combined and in side 12-145, the regular triangle which includes side 7-141 is combined. In a side of which end point is vertex 8 ~ 11, the regular triangle which includes the side of which end point is the same as the vertex is combined.



$S:(x,y)$      $S':(X',Y')$   
 $X'=x+6(n-1)-y$      $Y'=6(n-1)$   
 $jp1:(X'-1,Y'-1)$   
 $jp2:(X',Y'-1)$   
 side 7-141

Figure 7

In each figure, five regular triangles are combined in vertex. Therefore, two pixels among six pixels are the same, namely, skipping pixel exists. Namely, the number of pixels which are different is five. However, the repetition does not have influence on symmetrical painting by painting algorithm CW, CCW. Therefore, we can assume  $j_{max}$  of neighborhood array to be not 4 but 5.

### 3. Connection of icosahedrons(even tunnel)

If we connect two icosahedrons, the figure of Figure 8 is got. The red arrow is normal vector which represents front face of a surface.

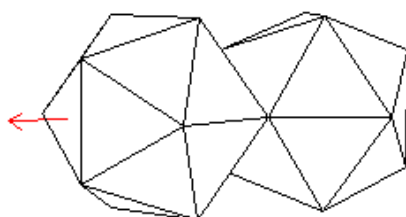


Figure 8

The two icosahedrons which have a hole touch on vertexes and sides which form the hole. The development is Figure 9.

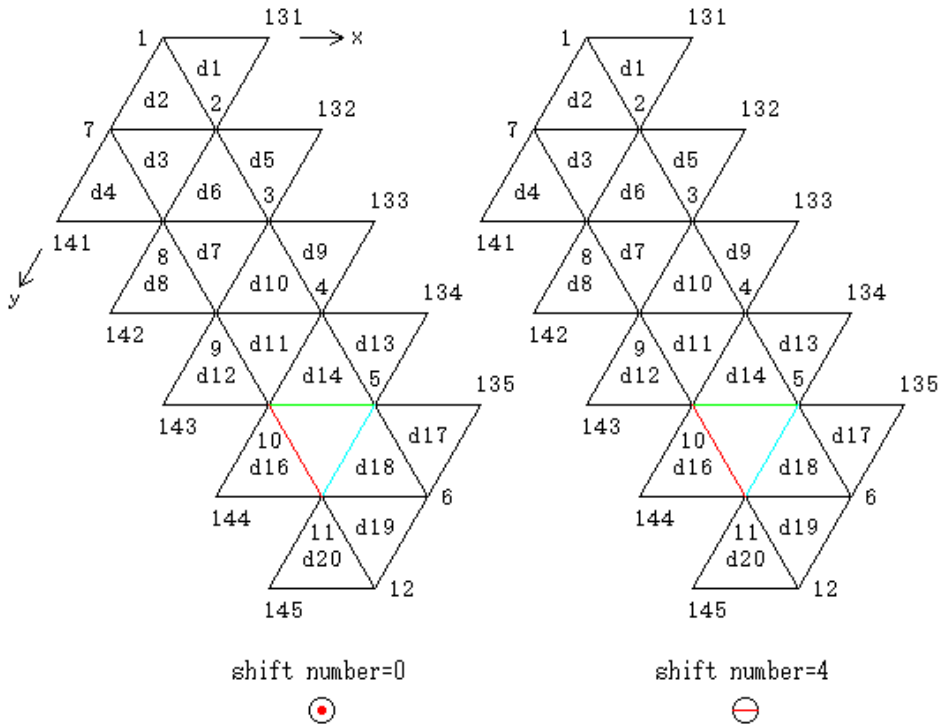


Figure 9

Neighborhood view on the side 10-11 of the left figure becomes like Figure 10.

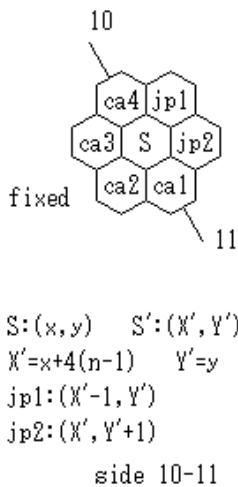


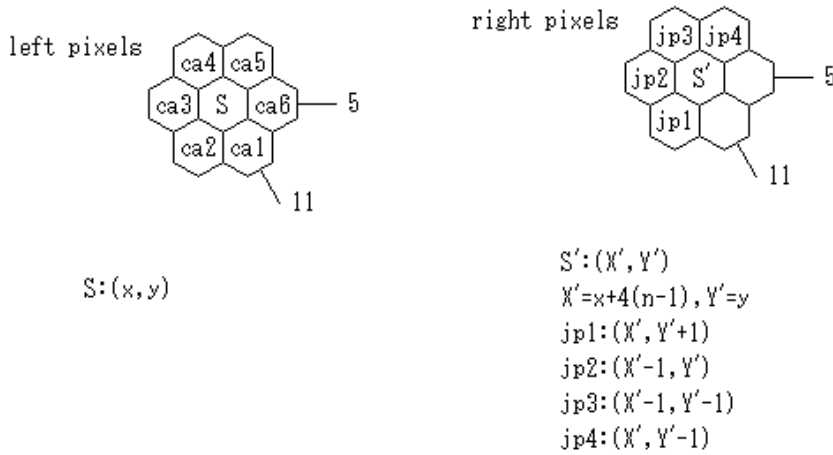
Figure 10

If jp1 or jp2 is chosen, the searching point S jumps to the right figure. The coordinates  $(X', Y')$  of the previous searching point S' after jump are

- $\Delta x = 4(n - 1)$
- $X' = x + \Delta x, Y' = y$

In Figure 9, If the left figure is moved by  $4(n - 1)$  in the direction of  $x$  axis, the right figure is got. Shift number is 4. If the same choice is done on the side 10-11 of the right figure, the searching point S jumps to the left figure and the previous searching point S' jumps by  $-\Delta x$ .

Neighborhood view on the vertex 10 of the left figure becomes like Figure 11.



vertex 10

Figure 11

We memorise two kinds of coordinates and shift number to neighborhood array in the order of base pixels ca6, ca5, ca4, ca3, ca2, ca1, additional pixels jp1, jp2, jp3, jp4.  $j_{max}$  becomes 9. If one of jp1, jp2, jp3, jp4 is chosen, the searching point S jumps to the right figure. The coordinates  $(X', Y')$  of the previous searching point S' after jump are

- $\Delta x = 4(n - 1)$
- $X' = x + \Delta x, Y' = y$

If the same choice is done on the vertex 10 of the right figure, the searching point S jumps to the left figure and the previous searching point S' jumps by  $-\Delta x$ .

The red symbols  $\cdot$  and  $-$  which are circled in Figure 9 represent front face and back face of a surface respectively. Whether front face or back face in Figure 10 in the last is the same as Figure 9. If we hope that all surfaces are colored after connecting the right figure and left figure, a device is needed. When we print out the graphics to paper, we must prepare both normal print out and reverse print out on right and left. On the left figure, we use the right half of reverse print out on right and left and on the right figure, we use the right half of normal print out. If the figure of Figure 9 is printed out to a transparent film, such distinction is unnecessary.

When we make a program, we consider the following conditions:

- (1) closes vertexes which form the hole; opens sides which form the hole.
- (2) opens vertexes which form the hole; closes sides which form the hole.
- (3) opens vertexes and sides which form the hole.

First, we make a program under (1) and if a symmetrical graphics is got, next, we choose one of the following steps:

- We make a program under (2) and if a symmetrical graphics is got, we make a program under (3).
- We make a program under (3).

#### 4. Connection of icosahedrons(odd tunnel)

If we connect three icosahedrons, the development is Figure 12.

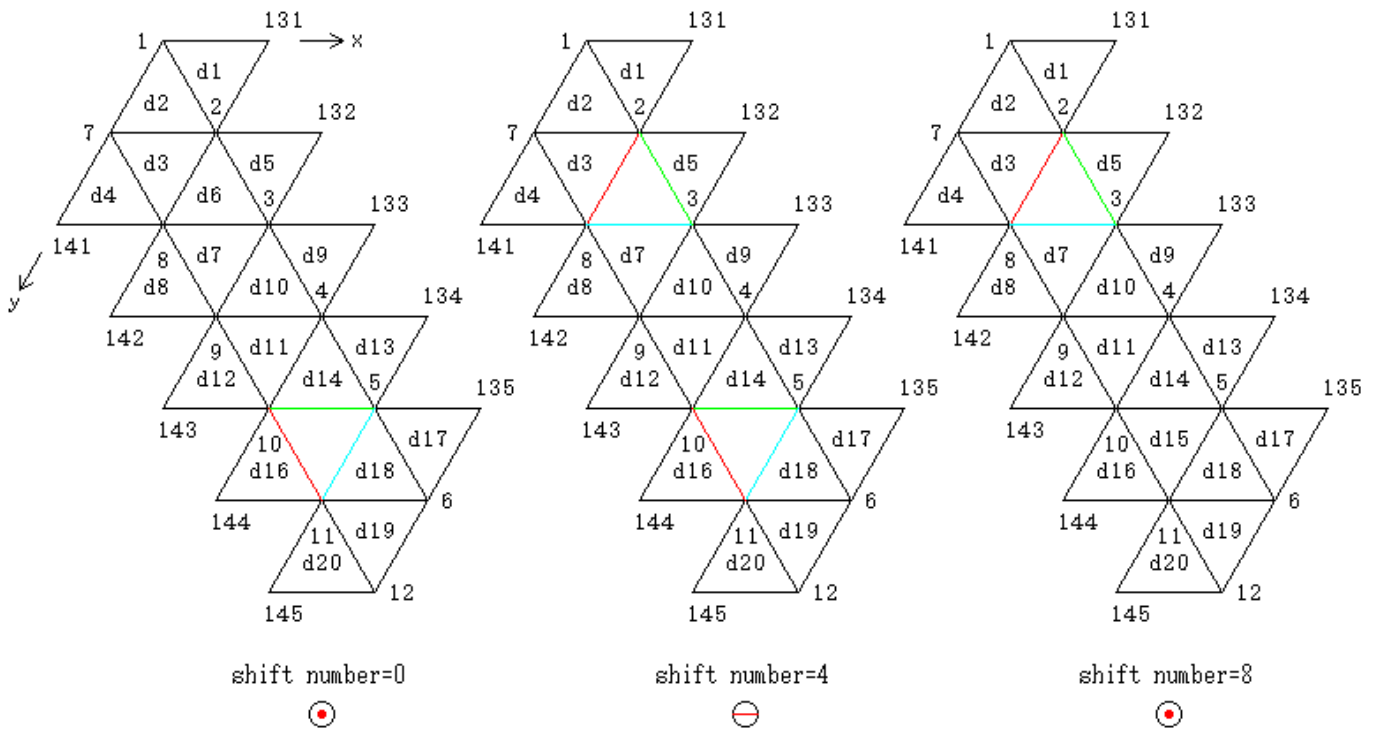


Figure 12

Whether front face or back face in Figure 13 in the last is the same as Figure 12.

Jump between shift number 0 and shift number 4 is done at vertexes and sides which form the lower hole and jump between shift number 4 and shift number 8 is done at vertexes and sides which form the upper hole.

### 5. Odd tunnel • Even tunnel

From the last but one, the name of odd tunnel and even tunnel has been used. The distinction between oddness and evenness is used on a graphics which is got by placing painting points which are divided in two on two convergence surfaces which are convergence surfaces each other. If a graphics is a combination of pieces which are one kind and a combination of pieces which are two kinds, it is called odd tunnel and even tunnel respectively. For the sake of ease, if final painting numbers are the same, we suppose that pieces are one kind.

We consider a plane which is parallel with a plane which is made by representative points of a minimum convergence surface and we project two minimum convergence surfaces on the plane. If two projected minimum convergence surfaces have a difference of rotation angle of  $180^\circ$  on the plane, a graphics becomes odd tunnel and If  $0^\circ$ , a graphics becomes even tunnel.

For example, If the number of octahedrons which are connected is one, because the difference is  $180^\circ$ , it becomes odd tunnel, if the number is two, because the difference is  $0^\circ$ , it becomes even tunnel and if the number is three, because the difference is  $180^\circ$ , it becomes odd tunnel. This circumstance is applied to  $5 \times 12$  polyhedron and icosahedron too. However, connection of only even tunnels cannot produce odd tunnel.

If a dihedron consists of two regular triangles, the difference is  $0^\circ$ . Therefore, it is even tunnel.

The number of regular triangles of a polyhedron can be represented using the number of regular triangles of a convergence surface. On octahedron,  $5 \times 12$  polyhedron, icosahedron, the numbers are resolved as follows:

$$\bullet 8 = (1 + 3) \times 2$$

- $60 = (5 + 5 \times 5) \times 2$
- $20 = (1 + 3 + 6) \times 2$

The first term, second term and third term are minimum convergence surface, second convergence surface and third convergence surface respectively.

## 6. Assignment

Complete neighborhood views on vertexes and sides of the figure of Figure 12.

## 7. Concrete example

Figure 13 is a symmetrical graphics by Figure 12 and the following are data of program.

- SP(3 × 2):3(left d6)+3(right d15):d?-c
- $n = 8$
- coordinates of painting number 1 : painting point  $a_0:(n - 1 + 1, n - 1 + 2)$ 、 painting point  $b_0:(n - 1 + 1, 2(n - 1) - 1)$ 、 painting point  $c_0:(2(n - 1) - 2, 2(n - 1) - 1)$
- coordinates of painting number 1 : painting point  $a'_1:(11(n - 1) + 2, 4(n - 1) + 1)$ 、 painting point  $b'_1:(12(n - 1) - 1, 5(n - 1) - 2)$ 、 painting point  $c'_1:(12(n - 1) - 1, 4(n - 1) + 1)$
- coordinates of painting number 2 : painting point  $a_0:\Delta y = 1$ 、 painting point  $b_0:\Delta x = 1$ 、 painting point  $c_0:\Delta x = -1;\Delta y = -1$
- coordinates of painting number 2 : painting point  $a'_1:\Delta x = 1;\Delta y = 1$ 、 painting point  $b'_1:\Delta y = -1$ 、 painting point  $c'_1:\Delta x = -1$

If painting number 2 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 : 15
- wall pixel : 0



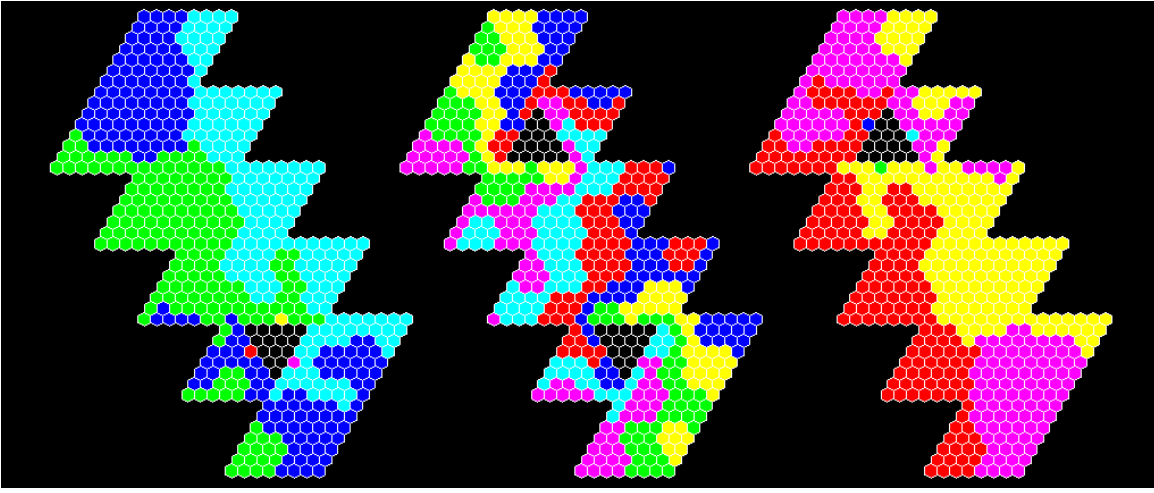


Figure 13

## セルラーオートマトングラフィクス (5)

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. 正 20 面体

図 1 は正 20 面体の展開図です。数字は頂点の番号であり、以下は対応する頂点、辺の組合せです。

- 131, 132, 133, 134, 135 ; 141, 142, 143, 144, 145 ; 1, 6 ; 7, 12 ;
- 2-131, 2-132 ; 3-132, 3-133 ; 4-133, 4-134 ; 5-134, 5-135 ; 1-131, 6-135;8-141, 8-142 ;
- 9-142, 9-143 ; 10-143, 10-144 ; 11-144, 11-145;7-141, 12-145;1-7, 6-12

頂点 2~5、8~11 を端点とする辺の対応は前回のカットと同じであり、辺 1-7、6-12 は平行移動です。

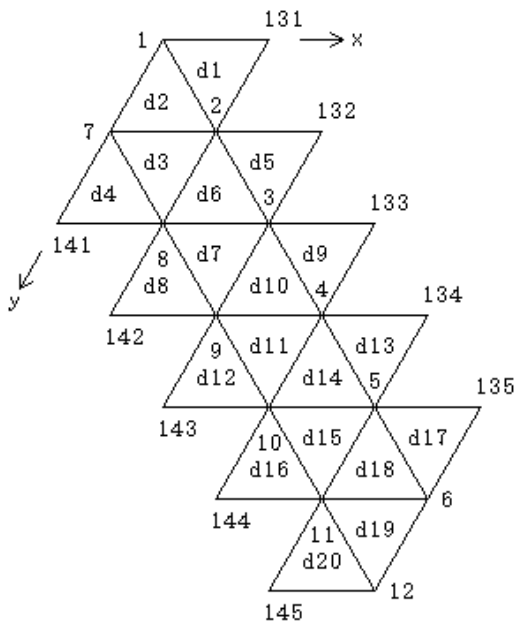
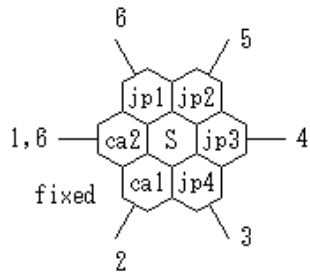


図 1

### 2. 近傍図

図 2 は頂点 131 に関する近傍図です。頂点 131 から頂点 135 ではこれらを含む五つの正三角形を結合させます。

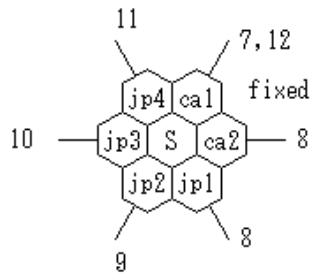


$S: (n-1, 0)$      $S': (X', Y')$   
 $X' = 5(n-1)$      $Y' = 4(n-1)$   
 $jp1: (X', Y'+1)$   
 $X' = 4(n-1)$      $Y' = 3(n-1)$   
 $jp2: (X', Y'+1)$   
 $X' = 3(n-1)$      $Y' = 2(n-1)$   
 $jp3: (X', Y'+1)$   
 $X' = 2(n-1)$      $Y' = n-1$   
 $jp4: (X', Y'+1)$

vertex 131

図 2

図 3 は頂点 141 に関する近傍図です。頂点 141 から頂点 145 ではこれらを含む五つの正三角形を結合させます。

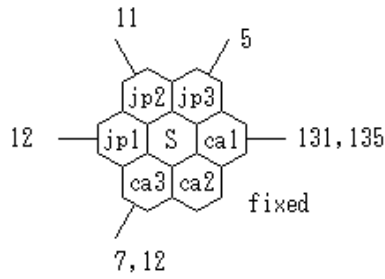


$S: (0, 2(n-1))$      $S': (X', Y')$   
 $X' = n-1$      $Y' = 3(n-1)$   
 $jp1: (X', Y'-1)$   
 $X' = 2(n-1)$      $Y' = 4(n-1)$   
 $jp2: (X', Y'-1)$   
 $X' = 3(n-1)$      $Y' = 5(n-1)$   
 $jp3: (X', Y'-1)$   
 $X' = 4(n-1)$      $Y' = 6(n-1)$   
 $jp4: (X', Y'-1)$

vertex 141

図 3

図 4 は頂点 1 に関する近傍図です。頂点 1 では頂点 1 の周りの二つの正三角形と頂点 6 の周りの三つの正三角形を、頂点 6 では頂点 6 の周りの三つの正三角形と頂点 1 の周りの二つの正三角形を結合させます。

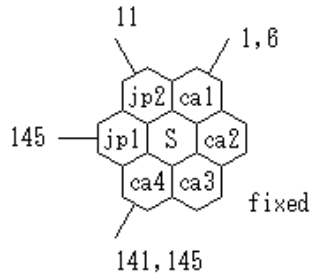


$S: (0, 0)$      $S': (X', Y')$   
 $X' = 5(n-1)$      $Y' = 5(n-1)$   
 $jp1: (X', Y'+1)$   
 $jp2: (X'-1, Y')$   
 $jp3: (X'-1, Y'-1)$

vertex 1

図 4

図 5 は頂点 7 に関する近傍図です。頂点 7 では頂点 7 の周りの三つの正三角形と頂点 12 の周りの二つの正三角形を、頂点 12 では頂点 12 の周りの二つの正三角形と頂点 7 の周りの三つの正三角形を結合させます。

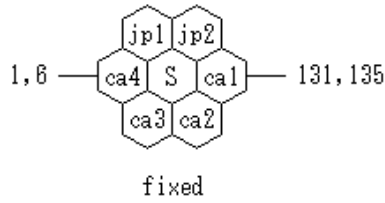


$S: (0, n-1)$      $S': (X', Y')$   
 $X' = 5(n-1)$      $Y' = 6(n-1)$   
 $jp1: (X'-1, Y')$   
 $jp2: (X'-1, Y'-1)$

vertex 7

図 5

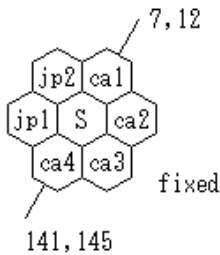
図 6 は辺 1-131 に関する近傍図です。辺の 2 端点は辺のピクセルには含めません。辺 1-131 では辺 6-135 を含む正三角形を、辺 6-135 では辺 1-131 を含む正三角形を結合させます。頂点 2~5 を端点とする辺では端点とその頂点と同じである辺を含む正三角形を結合させます。



$$\begin{aligned}
 S: (x, y) \quad S': (X', Y') \\
 X' = 5(n-1) \quad Y' = y + 5(n-1) - x \\
 jp1: (X'-1, Y') \\
 jp2: (X'-1, Y'-1) \\
 \text{side } 1-131
 \end{aligned}$$

図 6

図 7 は辺 7-141 に関する近傍図です。辺 7-141 では辺 12-145 を含む正三角形を、辺 12-145 では辺 7-141 を含む正三角形を結合させます。頂点 8~11 を端点とする辺では端点とその頂点と同じである辺を含む正三角形を結合させます。



$$\begin{aligned}
 S: (x, y) \quad S': (X', Y') \\
 X' = x + 6(n-1) - y \quad Y' = 6(n-1) \\
 jp1: (X'-1, Y'-1) \\
 jp2: (X', Y'-1) \\
 \text{side } 7-141
 \end{aligned}$$

図 7

各図において、頂点に関しては五つの正三角形を結合させています。したがって六つのピクセルの内二つは同じものであり、スキップピクセルが存在します。すなわち、異なるピクセルの数は五つです。ただし、この重複は塗りつぶしアルゴリズム CW、CCW による対称な描画には影響を及ぼしません。したがって近傍配列の  $j_{max}$  は 5 でかまいません。

### 3. 正 20 面体の連結 (偶トンネル)

正 20 面体を連結すると図 8 のようになります。赤い矢は面の表面を示す法線ベクトルです。

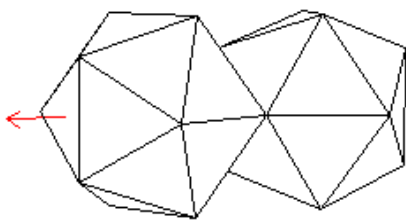


図 8

穴のあいた二つの正 20 面体は穴を形成する頂点と辺で接しています。展開図は図 9 です。

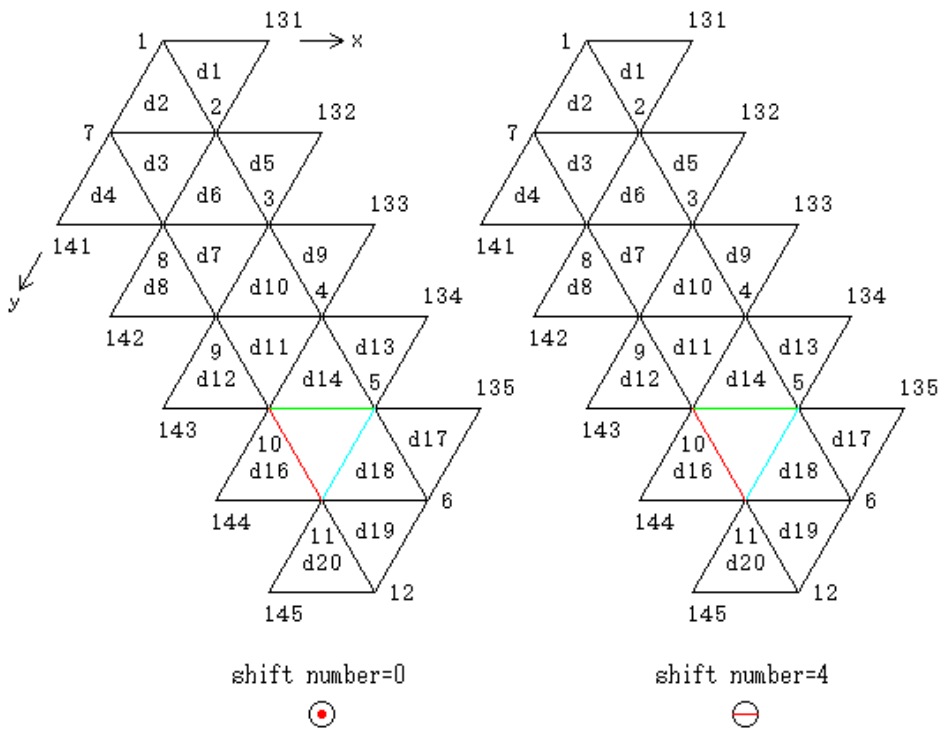
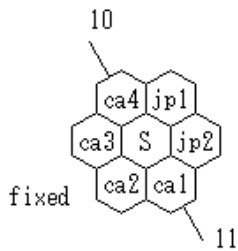


図 9

左の図形の辺 10-11 における近傍図は図 10 のようになります。



$$\begin{aligned}
 S:(x,y) & \quad S':(X',Y') \\
 X' &= x+4(n-1) \quad Y'=y \\
 jp1 &:(X'-1,Y') \\
 jp2 &:(X',Y'+1) \\
 & \text{side 10-11}
 \end{aligned}$$

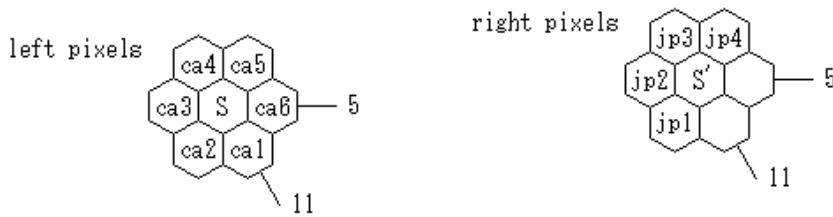
図 10

jp1 または jp2 が選択されれば探索点 S は右の図形にジャンプします。ジャンプ後の一つ前の探索点 S' の座標 (X', Y') は

- $\Delta x = 4(n - 1)$
- $X' = x + \Delta x$ 、 $Y' = y$

図 9 において、左の図形を  $x$  軸方向に  $4(n - 1)$  移動させると右の図形になります。シフト数は 4 です。右の図形の辺 10-11 において同様な選択がなされれば、探索点 S は左の図形にジャンプし、一つ前の探索点 S' は  $-\Delta x$  だけジャンプします。

左の図形の頂点 10 における近傍図は図 11 のようになります。



S:(x,y)

S':(X',Y')  
 $X'=x+4(n-1), Y'=y$   
 jp1:(X',Y'+1)  
 jp2:(X'-1,Y')  
 jp3:(X'-1,Y'-1)  
 jp4:(X',Y'-1)

vertex 10

図 11

近傍配列には二種類の座標とシフト数を、基礎ピクセル ca6、ca5、ca4、ca3、ca2、ca1、付加ピクセル jp1、jp2、jp3、jp4 の順に記憶します。j<sub>max</sub> は 9 となります。jp1、jp2、jp3、jp4 のいずれかが選択されれば探索点 S は右の図形にジャンプします。ジャンプ後の一つ前の探索点 S' の座標 (X', Y') は

- $\Delta x = 4(n - 1)$
- $X' = x + \Delta x, Y' = y$

右の図形の頂点 10 において同様な選択がなされれば、探索点 S は左の図形にジャンプし、一つ前の探索点 S' は  $-\Delta x$  だけジャンプします。

図 9 における で囲まれた赤い - と で囲まれた赤い・は各々表面、裏面を表します。前回の図 10 の表面、裏面は図 9 と同じです。左右の二つの図形を結合させて表面がすべて着色されているためには工夫が必要です。紙にプリントアウトする場合は通常プリントアウトと左右反転プリントアウトの両方を用意しなければなりません。左の図形は左右反転プリントアウトの右の図形を、右の図形は通常プリントアウトの右の図形を用います。透明フィルムへ図 9 の図形を出力すればこのような区別は無用です。

プログラムを作成する場合は下記の条件を考慮します。

- (1) 穴を形成する頂点を閉じる。穴を形成する辺を開く。
- (2) 穴を形成する頂点を開く。穴を形成する辺を閉じる。
- (3) 穴を形成する頂点と辺を開く。

最初は (1) の下でプログラムを作成し、対称グラフィクスが得られれば、次は下記のいずれかの手順を選択します。

- (2) の下でプログラムを作成し、対称グラフィクスが得られれば、(3) の下でプログラムを作成します。
- (3) の下でプログラムを作成します。

#### 4. 正 20 面体の連結 (奇トンネル)

正 20 面体を 3 連結した場合の展開図は図 12 です。

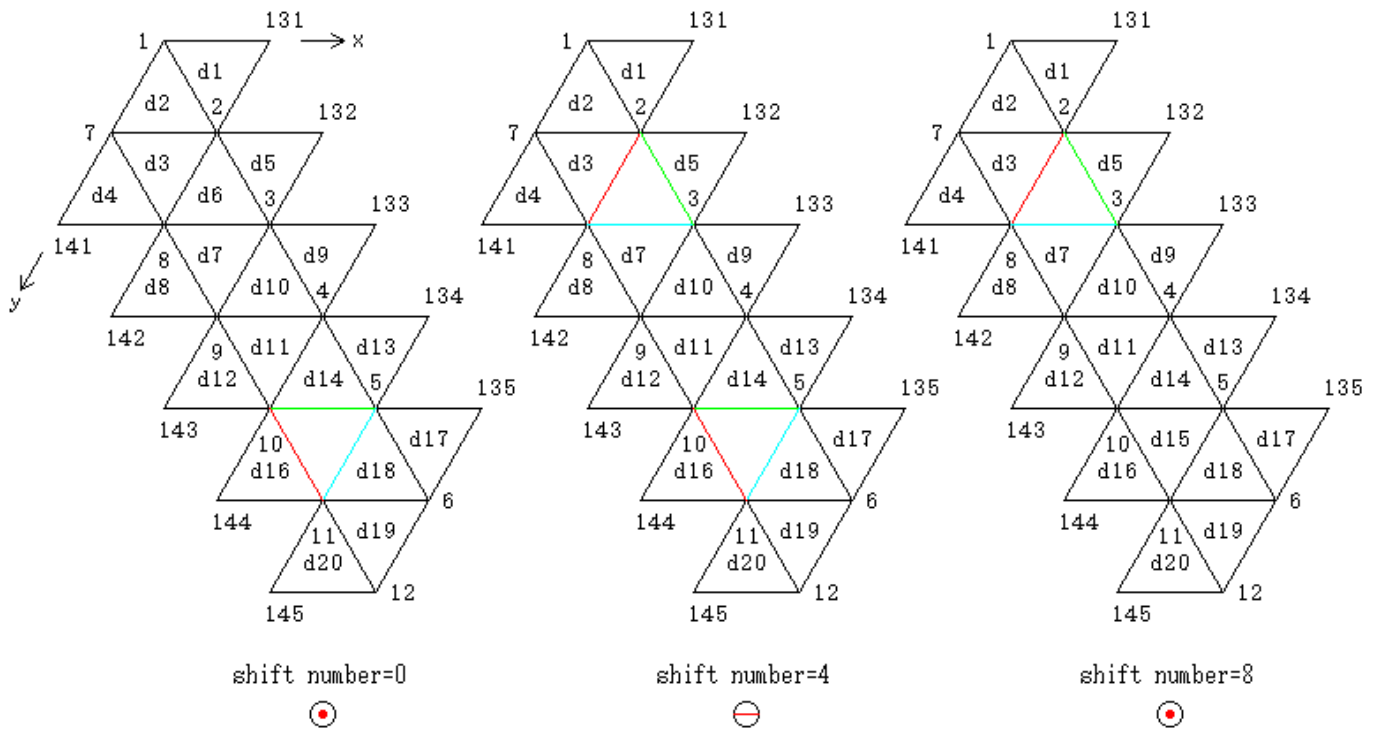


図 12

前回の図 13 の表面、裏面は図 12 と同じです。

シフト数 0 とシフト数 4 の間のジャンプは下の穴を形成する頂点と辺において、シフト数 4 とシフト数 8 の間のジャンプは上の穴を形成する頂点と辺において行います。

### 5. 奇トンネル・偶トンネル

前々回から奇トンネル、偶トンネルという名称を使ってきました。奇偶の区別は、互いに収束面である二つの収束面に等しい数の塗点を配して得られるグラフィクスに関して用いられます。グラフィクスが一種類のピースの組み合わせ、二種類のピースの組み合わせである場合に、このグラフィクスを各々奇トンネル、偶トンネルと称します。簡単のため、最終塗番号が同じであればピースが同種であるとしています。

最小収束面の代表点を作る平面に平行な平面を考え、二つの最小収束面をこの平面に投影します。もし二つの投影された最小収束面がこの平面上で、 $180^\circ$  の回転角の差があればグラフィクスは奇トンネルとなり、 $0^\circ$  の回転角の差があればグラフィクスは偶トンネルとなります。

たとえば、正八面体一つでは  $180^\circ$  の回転角の差ですから奇トンネルとなり、正八面体の 2 連結では  $0^\circ$  の回転角の差ですから偶トンネルとなり、正八面体の 3 連結では  $180^\circ$  の回転角の差ですから奇トンネルとなります。5 × 12 面体、正 20 面体においても事情は同じです。ただし、偶トンネルはいくつ連結しても偶トンネルです。

正二面体が二つの正三角形から成る場合は二つの正三角形の回転角の差は  $0^\circ$  です。したがって、この正二面体は偶トンネルです。

多面体の正三角形の数は収束面の正三角形の数を用いて表現することができます。正八面体、5 × 12 面体、正 20 面体では以下のようになります。

- $8 = (1 + 3) \times 2$
- $60 = (5 + 5 \times 5) \times 2$
- $20 = (1 + 3 + 6) \times 2$

初項は最小収束面、第二項は二次収束面、第三項は三次収束面です。



## 6. 課題

図 12 の図形の頂点、辺に関する近傍図を完成させてください。

## 7. 具体例

図 13 は図 12 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(3 × 2):3(left d6)+3(right d15):d?-c
- $n = 8$
- 塗番号 1 の座標 : 塗点  $a_0:(n - 1 + 1, n - 1 + 2)$ 、塗点  $b_0:(n - 1 + 1, 2(n - 1) - 1)$ 、塗点  $c_0:(2(n - 1) - 2, 2(n - 1) - 1)$
- 塗番号 1 の座標 : 塗点  $a'_1:(11(n - 1) + 2, 4(n - 1) + 1)$ 、塗点  $b'_1:(12(n - 1) - 1, 5(n - 1) - 2)$ 、塗点  $c'_1:(12(n - 1) - 1, 4(n - 1) + 1)$
- 塗番号 2 の座標 : 塗点  $a_0:\Delta y = 1$ 、塗点  $b_0:\Delta x = 1$ 、塗点  $c_0:\Delta x = -1;\Delta y = -1$
- 塗番号 2 の座標 : 塗点  $a'_1:\Delta x = 1;\Delta y = 1$ 、塗点  $b'_1:\Delta y = -1$ 、塗点  $c'_1:\Delta x = -1$

最初のグラフィクスで塗番号 2 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 : 15
- 壁ピクセル : 0

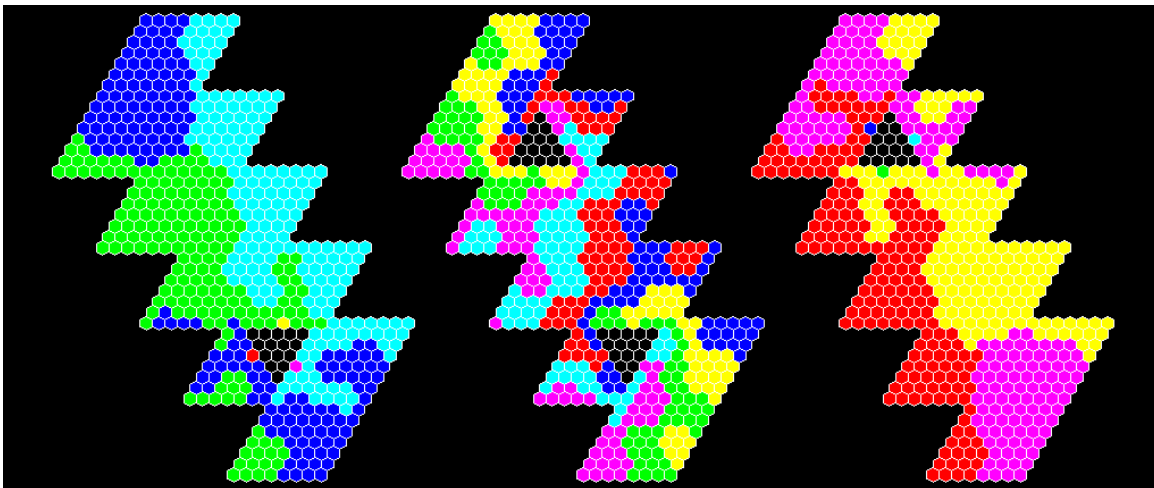


図 13

\*\*\*\*\*

List 1:cag\_5.c

```
/* t2.27 */
/* 2019 Morio Kikuchi */

#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

#define VGACOLORS (16+8)
#define GKS GetKeyState
#define ASIZE_MS ((1024*768)/CPMAX)
#define CPMAX /*3*//*6*/6
#define CPHALF (CPMAX/2)
#define X0 (120)
#define Y0 (10)
#define dyMAX 603
#define Odd /*0*//*1*//*3*/3
#define RESO /*8*//*10*//*12*/8 /* 9 for Odd:0 */
#define GRPH_0_MAX 10000

#define ICEIL(a,b) (((a)+((b)-1))/(b))

char refill,pauseflag,fieldflag,GRPH;
char charcode,charflag;
int X,Y,X_,Y_;
int enX[6+6],enY[6+6],enX_[6+6],enY_[6+6],enSN[6+6];
int algo,combination,drn,ig,PIXSIZE,dy_hex,jmpflag,sn_,sn;
int xt,yt,ssize,std_x,std_y,last_x,last_y,tmp0,tmp1;
long asize=ASIZE_MS;
long rcount[CPMAX],cnt;

/*unsigned */char **pixel;
long fp_mem[CPMAX];

char function,usflag;
unsigned char yorn;
int XRESO,YRESO,WB,DX_FRAME,DY_FRAME,DY_CAPTION;
FILE *fp;

typedef struct {int xx,yy,xx_,yy_,sn;} ss;
ss s;ss rtn[CPMAX][ASIZE_MS];
```

```

typedef struct {
unsigned char red,green,blue;} srgb;
srgb irgb[VGACOLORS];
typedef struct {
unsigned long back_;int back,fore;} bf;
bf bfset[]={WHITENESS,15,0},{BLACKNESS,0,15}};

HINSTANCE hinstance;
HWND hwnd;
HDC hdcdisplay,hdctmp1;
HBITMAP hbitmap1;
HPEN hpen;
HBRUSH hbrush;

void closegraph_(void),initpalette(void),BitBlt_full(void),setup(void),
cleardevice_(char,int,int,int,int),rectangle_(int,int,int,int),
delay_(long),beep(long),kbhit_(void),restore_3(void),initgraph_return(void),
use_subroop(void),keydowns_f2(void),bitblt(char,int,int,int,int,int,int),
arrayreset(void),fwrite_mem(int),fread_mem(int),putpixel_(int,int,int),
check_rcount(void),field(int,int,int);
unsigned char subroop(void);
int initgraph_(void),setup_(void),fourfloor_fiveceil(double),random_(int),
getpixel_(int,int,int,int),cag_r(void);
long ftell_mem(int);
double getangle(int,int);

COLORREF PALETTE(int color);
LRESULT CALLBACK wndproc_by_kbhit_(HWND,UINT,WPARAM,LPARAM);
int wndproc_filer(HWND,UINT,WPARAM,LPARAM);

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

if(argc>1 && strcmp(argv[1],"0")==0){
GRPH=0;
if(argc==2) argc=1;else argc=2;
}
else GRPH=1;
WB=1;
refill=1;
/*unlink("cpage.bin");*/

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

```

```

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

cnt=-1;
field(0,0,8);
field(0,0,4);
field(0,0,0);
cnt=0;

printf(" xt=%d yt=%d\n",xt/(RESO-1),yt/(RESO-1));
if(setup_()==1) return 1;
arrayreset();

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

combination=1;
drn=4;
val=1;                                     /* 0:no hall, 1:15, 2:8 */

while(1){
    field(val,0,8);
    if(Odd==3) field(val,val,4); /* for odd */
    else      field(0,val,4); /* for even */
    if(Odd!=1) field(0,val,0);
    else      field(0,0,0);
    cag_r();
    if(refill==0) break;
    check_rcount();
    printf(" \n");
    if(refill==0) break;

    if(GRPH){
    beep(50);

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

closegraph_();

return 0;

```

```

}/** main **/

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(flag==0){
}
else if(flag==1){
}
else if(flag==2){
}
else if(flag==3){
}
else return;

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);
}

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

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(hbitmap);
}
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 **/

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 **/

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*/)*(sqrt(3)/2)*PIXSIZE+10;
ls_image(2,"ss.bmp",0,0,XRESO,/*YRESO*/dy);
/*printf(" %d\n",dy);*/
beep(300);
}
}/** keydowns_f2 **/

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

bitblt(1,0,0,XRESO,YRESO,0,0);
}/** restore_in_PAINT **/

void setup(void)
{
XRESO=1024-4;YRESO=768-24*2;
}/** setup **/

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

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

return dx;
}/** get_dx **/

```



```

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

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

return dy;
}/** get_dy **/

int setup_(void)
{
int i,dy;

PIXSIZE=11;
dy=Y0+(yt+1/*1.7*/)*(sqrt(3)/2)*PIXSIZE+10;

if(dy>dyMAX){
while(1){
PIXSIZE--;
dy=Y0+(yt+1/*1.7*/)*(sqrt(3)/2)*PIXSIZE+10;
if(dy<=dyMAX) break;
}
}

if(PIXSIZE<4) PIXSIZE=4;

dy=get_dy(1,1)-get_dy(0,0);
dy_hex=ff_fc(dy/3.);

pixel=(*unsigned */char **)malloc(sizeof(*unsigned */char *)*((xt+1)+1));
if(pixel==NULL){
DeleteDC(hdctmp1);
DeleteObject(hbitmap1);
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);
DeleteDC(hdctmp1);
DeleteObject(hbitmap1);
initgraph_return();return 1;}

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

return 0;
}/** setup_ */

int initgraph_(void)
{
int i,width,height;
WNDCLASS wndclass;

setup();

wndclass.hInstance =hinstance;
wndclass.lpszClassName="CAGCLASS";
wndclass.lpszMenuName =NULL;
wndclass.lpfWndProc =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+DX_FRAME,YRESO+DY_CAPTION+DY_FRAME,
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);
hdctmp1=CreateCompatibleDC(hdcdisplay); /* text, dialog, menu */
SelectObject(hdctmp1,hbitmap1);
SetBkMode(hdcdisplay,TRANSPARENT);
SetBkMode(hdctmp1,TRANSPARENT);

initpalette();

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

return 0;
}/** initgraph_ **/

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

MessageBox(NULL,"Memory space is not left. ","CAG",MB_OK);
}/** initgraph_return **/

void closegraph_(void)
{
int i;

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

DeleteDC(hdctmp1);
DeleteObject(hbitmap1);

/*EndPaint(hwnd,&paintstruct);*/
ReleaseDC(hwnd,hdcdisplay);

```

```

DestroyWindow(hwnd);
/*UnregisterClass("CAGCLASS",hinstance);*/
}/** 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=1;i<7;i++){ /* 1 -> 6 */
if(irgb[i+8].red==255)
irgb[i].red=127+64;else irgb[i].red=0;
if(irgb[i+8].green==255)
irgb[i].green=127+64;else irgb[i].green=0;
if(irgb[i+8].blue==255)
irgb[i].blue=127+64;else irgb[i].blue=0;
}

for(i=7;i<9;i++){ /* 7, 8 */
irgb[i].red=127+32*(8-i);
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}

for(i=16;i<16+8;i++){ /* 16+8 colors */
irgb[i].red=255*(8-(i-16))/9;
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}
}/** initpalette */

void BitBlt_full(void)
{
bitblt(1,0,0,XRES0,YRES0,0,0);

```

```

}/** BitBlt_full **/

void bitblt(char flag,int x,int y,int xsize,int ysize,int x_,int y_)
{
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
        hdctmp1,x,y,SRCCOPY);
}/** bitblt **/

void cleardevice_(char flag,int x,int y,int xsize,int ysize)
{
PatBlt(hdctmp1,x,y,xsize,ysize,bfset[WB].back_);
}/** cleardevice_ **/

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

void kbhit_(void)
{
MSG msg;

if(PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
TranslateMessage(&msg);
DispatchMessage(&msg);
}
}/** kbhit_ */

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_ **/

int wndproc_filer(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
if(umsg==WM_KEYDOWN){
/***** menu keydowns -> *****/
/***** <- menu keydowns *****/

```

```

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

if(function==2){
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_CLOSE){
if(function==2) charflag=0;
else refill=0;

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

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

return 0;
}/** wndproc_filer **/

void delay_(long millisecond)
{
long oldtime,nowtime,dtime;
double 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;

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

void beep(long millisecond)
{
Beep(888,millisecond);
}/** beep **/

int fourfloor_fiveceil(double val_d)
{
int val_i,val;

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

return val;
}/** fourfloor_fiveceil **/

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

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 **/

int putpixel(int nx,int ny,int pcolor)
{
int i,dx,dy;
POINT vertex[7];

if(cnt<0) return 1;
if(nx<0 || ny<0) return 0;
if(GRPH==0) goto end;

dx=get_dx(nx,ny);
dy=get_dy(nx,ny);

vertex[0].x=dx;                vertex[0].y=dy;
vertex[2].x=get_dx(nx+1,ny+0);vertex[2].y=get_dy(nx+1,ny+0);
vertex[4].x=get_dx(nx+1,ny+1);vertex[4].y=get_dy(nx+1,ny+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;

if(pcolor==15)
hpen=CreatePen(PS_SOLID,1,PALETTE(9));
else
hpen=CreatePen(PS_SOLID,1,PALETTE(15));

/*if(nx==0 || ny==0 || nx==RESO-1 || ny==RESO-1) pcolor=12;*/
hbrush=CreateSolidBrush(PALETTE(pcolor));

SelectObject(hdcdisplay,hpen);
SelectObject(hdcdisplay,hbrush);

Polyline(hdcdisplay,vertex,6+1);
Polygon(hdcdisplay,vertex,6);

SelectObject(hdctmp1,hpen);
SelectObject(hdctmp1,hbrush);

```



```

Polyline(hdctmp1,vertex,6+1);
Polygon(hdctmp1,vertex,6);

DeleteObject(hbrush);
DeleteObject(hpen);

end:
pixel[nx][ny]=pcolor;

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(Odd>0 || CPMAX==3){
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)**/
}

```

```

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

```

```

void field_4(int x,int y,int dx,int dy)
{
int i,j;

```

```

for(j=y;j<y+dy;j++)
for(i=x;i<x+dx;i++){
kbhit_();
if(refill==0) break;
putpixel_(i,j,15);
if(i>xt) xt=i;
if(j>yt) yt=j;
}
}/** field_4 **/

```

```

void putdelta(int n,int lr,int x,int y,int dx,int dy,int dw)
{
int i,j;

if(lr==0){
for(j=0;j<n-1;j++)
for(i=0;i<j+dw;i++)
putpixel_(x+dx+i,y+dy+j,0);
}
else{
for(j=0;j<n-1;j++)
for(i=0;i<j+dw;i++)
putpixel_(x+dx+(n-2)-i,y+dy+(n-2)-j,0);
}
}/** putdelta **/

```

```

void field(int upper,int lower,int begin)
{
int i,n,XDP,YDP,ds;

fieldflag=1;

n=RESO;
ds=n-1;
sn=begin;

for(i=0;i<5;i++)
field_4(begin*(RESO-1)+(RESO-1)*i,(RESO-1)*i,RESO,2*RESO-1);

if(upper>0) putdelta(RESO,0,(sn+1)*(RESO-1),1*(RESO-1),1,0,-1);
if(upper==2){
putpixel(1*(n-1)+sn*(n-1),1*(n-1),8);
putpixel(1*(n-1)+sn*(n-1),2*(n-1),8);
putpixel(2*(n-1)+sn*(n-1),2*(n-1),8);
}
if(lower>0) putdelta(RESO,1,(sn+3)*(RESO-1),4*(RESO-1),0,1,-1);
if(lower==2){
putpixel(3*(n-1)+sn*(n-1),4*(n-1),8);
putpixel(4*(n-1)+sn*(n-1),4*(n-1),8);
putpixel(4*(n-1)+sn*(n-1),5*(n-1),8);
}

if(cnt>-1 && n%3==1){
i=1;
XDP=n/3+ds*i+ds*0;YDP=n-(n/3+1)+ds*i;pixel[XDP][YDP]=-1;
if(Odd==1){
i=3;
XDP=n-(n/3+1)+ds*i+ds*0;YDP=n/3+ds+ds*i;pixel[XDP][YDP]=-1;
}

if(Odd==0){
i=1;
XDP=n/3+ds*i+ds*4;YDP=n-(n/3+1)+ds*i;pixel[XDP][YDP]=-1;
/*i=3;
XDP=n-(n/3+1)+ds*i+ds*4;YDP=n/3+ds+ds*i;pixel[XDP][YDP]=-1;*/
}
else if(Odd==3){
/*i=1;
XDP=n/3+ds*i+ds*8;YDP=n-(n/3+1)+ds*i;pixel[XDP][YDP]=-1;*/
i=3;

```

```
XDP=n-(n/3+1)+ds*i+ds*8;YDP=n/3+ds+ds*i;pixel[XDP][YDP]=-1;
}
}
```

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

```
void pp_(int nx,int ny,int pcolor)
{
nx+=sn;
nx*=RESO-1;ny*=RESO-1;
putpixel(nx,ny,pcolor);
}/** pp_ **/
```

```
void rot(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 **/
```

```
int nh(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 **/

```

```

int _13(int flag,int x,int y,int nx,int ny,int pcolor)
{
int i,n,dlt;

```

```

n=RESO;

```

```

if(flag<2){
for(i=0;i<5;i++){
if(x==(n-1)*(i+1) && y==(n-1)*i){
if(flag==0) return 1;
else{
if(i==0){
/* 131 */

```

```

        if(nx==x+1 && ny==y+1) {dlt=n-1;X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y) {dlt=2*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x && ny==y-1) {dlt=3*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx>(i+1)*(n-1) || ny<i*(n-1)) return -1;
}

```

```

else if(i==1){
/* 132 */

```

```

        if(nx==x+1 && ny==y+1) {dlt=n-1;X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y) {dlt=2*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x && ny==y-1) {dlt=3*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx>(i+1)*(n-1) || ny<i*(n-1)) return -1;
}

```

```

else if(i==2){
/* 133 */

```

```

        if(nx==x+1 && ny==y+1) {dlt=n-1;X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y) {dlt=2*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}

```

```

else if(nx==x && ny==y-1) {dlt=-2*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx>(i+1)*(n-1) || ny<i*(n-1)) return -1;
}
else if(i==3){
/* 134 */

    if(nx==x+1 && ny==y+1) {dlt=n-1;X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y) {dlt=-3*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x && ny==y-1) {dlt=-2*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx>(i+1)*(n-1) || ny<i*(n-1)) return -1;
}
else if(i==4){
/* 135 */

    if(nx==x+1 && ny==y+1) {dlt=-4*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y) {dlt=-3*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x && ny==y-1) {dlt=-2*(n-1);X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx>(i+1)*(n-1) || ny<i*(n-1)) return -1;

/*if(cnt==103 && rcount[2]>=66 && ig==2){
printf(" %ld %d\n",rcount[2],dlt);
}*/
}

return 1;
}/**else(flag,0)**/
}/**if(x,y)**/
}/**for(i)**/
}/**if(flag,2)**/
else{
for(i=0;i<5;i++){
if(x!=(n-1)*(i+1) || y!=(n-1)*i)
putpixel((n-1)*(i+1)+sn*(n-1),(n-1)*i,pcolor);
}
}

return 0;
}/** _13 **/

int _14(int flag,int x,int y,int nx,int ny,int pcolor)
{
int i,n,dlt;

n=RESO;

```



```

return 1;
}/**else(flag,0)**/
}/**if(x,y)**/
}/**for(i)**/
}/**if(flag,2)**/
else{
for(i=0;i<5;i++){
if(x!=(n-1)*i || y!=(n-1)*(i+2))
putpixel((n-1)*i+sn*(n-1),(n-1)*(i+2),pcolor);
}
}

return 0;
}/** _14 **/

```

```

int bside(int x,int y,int nx,int ny)
{
int n,dlt;

n=RES0;
dlt=0;

if(x==n-1 && y>=1 && y<=n-2){
/* 2(left) */
dlt=n-1-y;
if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y) {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}
else if(y==n-1 && x>=n && x<=2*n-3){
/* 2(right) */
dlt=n-1-x;
if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_;jmpflag=3;}
else if(nx==x && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}
else if(x==2*(n-1) && y>=n && y<=2*n-3){
/* 3(left) */

dlt=2*(n-1)-y;
if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y) {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}
else if(y==2*(n-1) && x>=2*n-1 && x<=3*n-4){
/* 3(right) */
dlt=2*(n-1)-x;
if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_;jmpflag=3;}

```



```

else if(nx==x && ny==y-1)    {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}
else if(x==3*(n-1) && y>=2*n-1 && y<=3*n-4){
/* 4(left) */
dlt=3*(n-1)-y;
    if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y)    {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}
else if(y==3*(n-1) && x>=3*n-2 && x<=4*n-5){
/* 4(right) */
dlt=3*(n-1)-x;
    if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_;jmpflag=3;}
else if(nx==x && ny==y-1)  {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}
else if(x==4*(n-1) && y>=3*n-2 && y<=4*n-5){
/* 5(left) */
dlt=4*(n-1)-y;
    if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y)  {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}
else if(y==4*(n-1) && x>=4*n-3 && x<=5*n-6){
/* 5(right) */
dlt=4*(n-1)-x;
    if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_;jmpflag=3;}
else if(nx==x && ny==y-1)  {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}

else if(y==2*(n-1) && x>=1 && x<=n-2){
/* 8(left) */
dlt=n-1-x;
    if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_;jmpflag=3;}
else if(nx==x && ny==y+1)  {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}
else if(x==n-1 && y>=2*n-1 && y<=3*n-4){
/* 8(right) */
dlt=2*(n-1)-y;
    if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_-1;jmpflag=3;}
else if(nx==x-1 && ny==y)  {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}
else if(y==3*(n-1) && x>=n && x<=2*n-3){
/* 9(left) */
dlt=2*(n-1)-x;
    if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_;jmpflag=3;}
else if(nx==x && ny==y+1)  {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}

```

```

else if(x==2*(n-1) && y>=3*n-2 && y<=4*n-5){
/* 9(right) */
dlt=3*(n-1)-y;
    if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_-1;jmpflag=3;}
else if(nx==x-1 && ny==y)    {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}
else if(y==4*(n-1) && x>=2*n-1 && x<=3*n-4){
/* 10(left) */
dlt=3*(n-1)-x;
    if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_;jmpflag=3;}
else if(nx==x && ny==y+1)  {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}
else if(x==3*(n-1) && y>=4*n-3 && y<=5*n-6){
/* 10(right) */
dlt=4*(n-1)-y;
    if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_-1;jmpflag=3;}
else if(nx==x-1 && ny==y)  {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}
else if(y==5*(n-1) && x>=3*n-2 && x<=4*n-5){
/* 11(left) */
dlt=4*(n-1)-x;
    if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_;jmpflag=3;}
else if(nx==x && ny==y+1)  {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}
else if(x==4*(n-1) && y>=5*n-4 && y<=6*n-7){
/* 11(right) */
dlt=5*(n-1)-y;
    if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_-1;jmpflag=3;}
else if(nx==x-1 && ny==y)  {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}

else if(y==0 && x>=1 && x<=n-2){
/* 1-131 */
dlt=5*(n-1)-x;
    if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_;jmpflag=3;}
else if(nx==x && ny==y-1)  {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}
else if(x==5*(n-1) && y>=4*n-3 && y<=5*n-6){
/* 6-135 */
dlt=0-y;
    if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_+1;jmpflag=3;}
else if(nx==x+1 && ny==y)  {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}

```

```

else if(x==0 && y>=n && y<=2*n-3){
/* 7-141 */
dlt=6*(n-1)-y;
    if(nx==x-1 && ny==y-1) {X_=x+dlt;Y_=y+dlt;X=X_;Y=Y_-1;jmpflag=3;}
else if(nx==x-1 && ny==y)    {X_=x+dlt;Y_=y+dlt;X=X_-1;Y=Y_-1;jmpflag=3;}
}
else if(y==6*(n-1) && x>=4*n-3 && x<=5*n-6){
/* 12-145 */
dlt=0-x;
    if(nx==x+1 && ny==y+1) {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_;jmpflag=3;}
else if(nx==x && ny==y+1)  {X_=x+dlt;Y_=y+dlt;X=X_+1;Y=Y_+1;jmpflag=3;}
}

return dlt;
}/** bside **/

int side(int ft,int t,int side,int nx,int ny)
{
int n;

n=RES0;

if(ft==4 && t==3 && side==0){
if(nx>=3*(n-1)+1 && nx<=4*(n-1)-1 && ny==4*(n-1)) return 1;else return 0;
}
else if(ft==4 && t==3 && side==1){
if(nx==4*(n-1) && ny>=4*(n-1)+1 && /*nx*/ny<=5*(n-1)-1) return 1;else return 0;
}
else if(ft==4 && t==3 && side==2){
if(nx>=3*(n-1)+1 && nx<=4*(n-1)-1 && ny==nx+1*(n-1)) return 1;else return 0;
}
else if(ft==2 && t==2 && side==0){
if(nx==1*(n-1) && ny>=1*(n-1)+1 && ny<=2*(n-1)-1) return 1;else return 0;
}
else if(ft==2 && t==2 && side==1){
if(nx>=1*(n-1)+1 && nx<=2*(n-1)-1 && ny==2*(n-1)) return 1;else return 0;
}
else if(ft==2 && t==2 && side==2){
if(nx>=1*(n-1)+1 && nx<=2*(n-1)-1 && ny==nx+0*(n-1)) return 1;else return 0;
}

return 0;
}/** side **/

```

```

int v(int ft,int t,int v,int nx,int ny)
{
int n;

n=RESO;

if(ft==4 && t==3 && v==0){
if(nx==3*(n-1) && ny==4*(n-1)) return 1;else return 0;
}
else if(ft==4 && t==3 && v==1){
if(nx==4*(n-1) && ny==4*(n-1)) return 1;else return 0;
}
else if(ft==4 && t==3 && v==2){
if(nx==4*(n-1) && ny==5*(n-1)) return 1;else return 0;
}

else if(ft==2 && t==2 && v==0){
if(nx==1*(n-1) && ny==1*(n-1)) return 1;else return 0;
}
else if(ft==2 && t==2 && v==1){
if(nx==1*(n-1) && ny==2*(n-1)) return 1;else return 0;
}
else if(ft==2 && t==2 && v==2){
if(nx==2*(n-1) && ny==2*(n-1)) return 1;else return 0;
}

return 0;
}/** v **/

```

```

void putpixel_(int nx,int ny,int pcolor)
{
int n,dlt,flag,snold;

n=RESO;

putpixel(nx,ny,pcolor);
if(fieldflag) return;
rcount[ig]++;

/*return;*/ /* here */
nx-=sn*(n-1);

if(nx==0 && ny==0){
/* 1 */
putpixel(5*(n-1)+sn*(n-1),5*(n-1),pcolor/*-1*/);
}

```

```

}
else if(nx==5*(n-1) && ny==5*(n-1)){
/* 6 */
nx+=sn*(n-1);
putpixel(0+sn*(n-1),0,pcolor/*-1*/);
}
else if(nx==0 && ny==n-1){
/* 7 */
nx+=sn*(n-1);
putpixel(5*(n-1)+sn*(n-1),6*(n-1),pcolor/*-1*/);
}
else if(nx==5*(n-1) && ny==6*(n-1)){
/* 12 */
nx+=sn*(n-1);
putpixel(0+sn*(n-1),n-1,pcolor/*-1*/);
}
else if(_13(0,nx,ny,-1,-1,-1)==1){
/* 13 */
_13(2,nx,ny,-1,-1,pcolor/*-1*/);
}
else if(_14(0,nx,ny,-1,-1,-1)==1){
/* 14 */
_14(2,nx,ny,-1,-1,pcolor/*-1*/);
}
else if(/*_2345(0,nx,ny,-1,-1)==1*/0){
/* 2345 */
if(0) putpixel(nx,ny,pcolor-1);
}
else if(/*_891011(0,nx,ny,-1,-1)==1*/0){
/* 891011 */
if(0) putpixel(nx,ny,pcolor-1);
}
/*999*/
#if Odd!=1
else if(v(4,3,0,nx,ny)==1 && sn<8){
snold=sn;
if(sn==0) sn=4;
else if(sn==4) sn=0;

pp_(3,4,pcolor);
sn=snold;
}
else if(v(4,3,1,nx,ny)==1 && sn<8){
snold=sn;
if(sn==0) sn=4;
else if(sn==4) sn=0;

```

```

pp_(4,4,pcolor);
sn=snold;
}
else if(v(4,3,2,nx,ny)==1 && sn<8){
snold=sn;
if(sn==0) sn=4;
else if(sn==4) sn=0;

pp_(4,5,pcolor);
sn=snold;
}
#endif

#if Odd==3
else if(v(2,2,0,nx,ny)==1 && sn>0){
snold=sn;
if(sn==4) sn=8;
else if(sn==8) sn=4;

pp_(1,1,pcolor);
sn=snold;
}
else if(v(2,2,1,nx,ny)==1 && sn>0){
snold=sn;
if(sn==4) sn=8;
else if(sn==8) sn=4;

pp_(1,2,pcolor);
sn=snold;
}
else if(v(2,2,2,nx,ny)==1 && sn>0){
snold=sn;
if(sn==4) sn=8;
else if(sn==8) sn=4;

pp_(2,2,pcolor);
sn=snold;
}
#endif
else{
if(/*ny==nx*/0) flag=0; /* 1-6 */
else if(/*ny==nx+n-1*/0) flag=0; /* 7-12 */
else if(nx==0 && ny>=1 && ny<=n-2) flag=1; /* 1-7 */
else if(nx==5*(n-1) && ny>=5*n-4 && ny<=6*n-7) flag=2; /* 6-12 */
else if(/*_2345_891011(nx,ny)==1*/0) flag=0; /* backslash */

```

```

else if( /*_23456_7891011(nx,ny)==1*/0) flag=0;           /* slash */
else if((dlt=bside(nx,ny,-1,-1))!=0)                    flag=3; /* border side */

#if Odd!=1
else if(side(4,3,0,nx,ny)==1 && sn<8) flag=119;
else if(side(4,3,1,nx,ny)==1 && sn<8) flag=120;
else if(side(4,3,2,nx,ny)==1 && sn<8) flag=121;
#endif

/* for odd */
#if Odd==3
else if(side(2,2,0,nx,ny)==1 && sn>0) flag=122;
else if(side(2,2,1,nx,ny)==1 && sn>0) flag=123;
else if(side(2,2,2,nx,ny)==1 && sn>0) flag=124;
#endif

else flag=-1;

if(flag>0){
if(flag==1){
nx+=5*(n-1);ny+=5*(n-1);
}
else if(flag==2){
nx-=5*(n-1);ny-=5*(n-1);
}
else if(flag==3){
nx+=dlt;ny+=dlt;
}

else if(flag>=119 && flag<=121){ /* ft4-t3 */
snold=sn;
if(sn==0) sn=4;
else if(sn==4) sn=0;
}
else if(flag>=122 && flag<=124){ /* ft2-t2 */
snold=sn;
if(sn==4) sn=8;
else if(sn==8) sn=4;
}

nx+=sn*(n-1);
putpixel(nx,ny,pcolor);

if(flag>=119 && flag<=124){

```

```

sn=snold;
}
}/**else if(flag>0)**/
}/**else(_891011)**/
}/** putpixel_ **/

int getpixel_(int x,int y,int nx,int ny)
{
int i,n,flag,val,dsn;

n=RESO;

x-=sn*(n-1);
nx-=sn*(n-1);

/*if(nx<0 || ny<0 || nx>5*(n-1) || ny>6*(n-1)) return 0;
for(i=1;i<=4;i++){
if(nx>i*(n-1) && ny<i*(n-1)) return 0;
if(nx<i*(n-1) && ny>(i+1)*(n-1)) return 0;
}*/ /* here */

X=nx;Y=ny;
X_=x;Y_=y;
jmpflag=0;
dsn=0;
flag=0;
/*goto end;*/ /* here */
/*if(ig==2 && rcount[ig]==25) printf(" %d %d %d %d\n",x,y,nx,ny);*/

if(x==0 && y==0){
/* 1 */

    if(nx==x && ny==y-1) {X_=x+5*(n-1);Y_=y+5*(n-1);X=X_-1;Y=Y_-1;jmpflag=1;}
else if(nx==x-1 && ny==y-1) {X_=x+5*(n-1);Y_=y+5*(n-1);X=X_-1;Y=Y_-1;jmpflag=1;}
else if(nx<0 || ny<0) return 0;
}
else if(x==5*(n-1) && y==5*(n-1)){
/* 6 */

    if(nx==x+1 && ny==y+1) {X_=x-5*(n-1);Y_=y-5*(n-1);X=X_+1;Y=Y_+1;jmpflag=2;}
else if(nx==x+1 && ny==y) {X_=x-5*(n-1);Y_=y-5*(n-1);X=X_+1;Y=Y_-1;jmpflag=2;}
else if(nx>5*(n-1)) return 0;
}
else if(x==0 && y==n-1){
/* 7 */

```



```

        if(nx==x-1 && ny==y-1) {X_=x+5*(n-1);Y_=y+5*(n-1);X=X_-1;Y=Y_-1;jmpflag=1;}
else if(nx==x-1 && ny==y) {X_=x+5*(n-1);Y_=y+5*(n-1);X=X_-1;Y=Y_;jmpflag=1;}
else if(nx<0) return 0;
}
else if(x==5*(n-1) && y==6*(n-1)){
/* 12 */

        if(nx==x && ny==y+1) {X_=x-5*(n-1);Y_=y-5*(n-1);X=X_+1;Y=Y_+1;jmpflag=2;}
else if(nx==x+1 && ny==y+1) {X_=x-5*(n-1);Y_=y-5*(n-1);X=X_+1;Y=Y_;jmpflag=2;}
else if(nx>5*(n-1) || ny>6*(n-1)) return 0;
}
else if(abs((val=_13(1,x,y,nx,ny,-1)))==1){
/* 13 */
if(val== -1) return 0;
}
else if(abs((val=_14(1,x,y,nx,ny,-1)))==1){
/* 14 */
if(val== -1) return 0;
}
else if(/*abs((val=_2345(1,x,y,nx,ny)))==1*/0){
/* 2345 */
if(val== -1) {printf(" ?\n");return 0;}
}
else if(/*abs((val=_891011(1,x,y,nx,ny)))==1*/0){
/* 891011 */
if(val== -1) {printf(" ?\n");return 0;}
}
else{
if(/*y==x*/0)                flag=0;                /* 1-6 */
else if(/*y==x+n-1*/0)        flag=0;                /* 7-12 */
else if(x==0 && y>=1 && y<=n-2)    flag=1;        /* 1-7 */
else if(x==5*(n-1) && y>=5*n-4 && y<=6*n-7) flag=2;    /* 6-12 */
else if(/*_2345_891011(x,y)==1*/0)  flag=0;                /* backslash */
else if(/*_23456_7891011(x,y)==1*/0) flag=0;                /* slash */
else if(bsite(x,y,nx,ny)!=0 && jmpflag==3)  flag=3;    /* border side */

#if Odd!=1
else if(side(4,3,0,x,y)==1 && sn<8) flag=119;
else if(side(4,3,1,x,y)==1 && sn<8) flag=120;
else if(side(4,3,2,x,y)==1 && sn<8) flag=121;
#endif

/* for odd */
#if Odd==3
else if(side(2,2,0,x,y)==1 && sn>0) flag=122;

```

```

else if(side(2,2,1,x,y)==1 && sn>0) flag=123;
else if(side(2,2,2,x,y)==1 && sn>0) flag=124;
#endif

else flag=0;

if(flag>0){
if(flag==1){
if(nx<0) {X+=5*(n-1);Y+=5*(n-1);X_+=5*(n-1);Y_+=5*(n-1);jmpflag=1;}
}
else if(flag==2){
if(nx>5*(n-1)) {X-=5*(n-1);Y-=5*(n-1);X_-=-5*(n-1);Y_-=-5*(n-1);jmpflag=2;}
}
else if(flag==3){
/*X_=-;Y_=-;X=;Y=;*/
}

else if(flag>=119 && flag<=121){ /* ft4-t3 */
if(flag==119){ /* side(4,3,0, */
if(nh(x,y,nx,ny,1)==1) {rot(1,-2);jmpflag=101;}
else if(nh(x,y,nx,ny,2)==1) {rot(2,2);jmpflag=101;}
if(jmpflag>100) {if(sn==0) dsn=4;else if(sn==4) dsn=-4;}
}
else if(flag==120){ /* side(4,3,1, */
if(nh(x,y,nx,ny,3)==1) {rot(3,-2);jmpflag=101;}
else if(nh(x,y,nx,ny,4)==1) {rot(4,2);jmpflag=101;}
if(jmpflag>100) {if(sn==0) dsn=4;else if(sn==4) dsn=-4;}
}
else if(flag==121){ /* side(4,3,2 */
if(nh(x,y,nx,ny,5)==1) {rot(5,-2);jmpflag=101;}
else if(nh(x,y,nx,ny,0)==1) {rot(0,2);jmpflag=101;}
if(jmpflag>100) {if(sn==0) dsn=4;else if(sn==4) dsn=-4;}
}
}
else if(flag>=122 && flag<=124){ /* ft2-t2 */
if(flag==122){ /* side(2,2,0, */
if(nh(x,y,nx,ny,0)==1) {rot(0,-2);jmpflag=101;}
else if(nh(x,y,nx,ny,1)==1) {rot(1,2);jmpflag=101;}
if(jmpflag>100) {if(sn==4) dsn=4;else if(sn==8) dsn=-4;}
}
else if(flag==123){ /* side(2,2,1, */
if(nh(x,y,nx,ny,4)==1) {rot(4,-2);jmpflag=101;}
else if(nh(x,y,nx,ny,5)==1) {rot(5,2);jmpflag=101;}
if(jmpflag>100) {if(sn==4) dsn=4;else if(sn==8) dsn=-4;}
}
}

```

```

}
else if(flag==124){ /* side(2,2,2 */
    if(nh(x,y,nx,ny,2)==1) {rot(2,-2);jmpflag=101;}
else if(nh(x,y,nx,ny,3)==1) {rot(3,2);jmpflag=101;}
if(jmpflag>100) {if(sn==4) dsn=4;else if(sn==8) dsn=-4;}
}
}

```

```

if(flag>=119 && flag<=124 && jmpflag>100){
X=x+tmp0;
Y=y+tmp1;
X_=x;
Y_=y;
}
}/**else if(flag>0)**/
}/**else(_891011)**/

```

```

/*if(ig==2 && side(4,3,2,nx,ny)==1){
if(side(4,3,1,x,y)==1) printf(" ? %d %d\n",x,y);
printf(" x:%d y:%d flag:%d dsn:%d\n",x,y,flag,dsn);
}*/

```

```

end:
X+=sn*(n-1); /* restore */
X_+=sn*(n-1);

```

```

X+=dsn*(n-1); /* jump */
X_+=dsn*(n-1);
sn_=sn+dsn; /* new sn */

```

```

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 **/

int check_v(int x,int y)
{
int val;

x-=sn*(RESO-1);

if(0) ;

#if Odd!=1
else if(v(4,3,0,x,y)==1 && sn<8) val=1;

```

```

else if(v(4,3,1,x,y)==1 && sn<8) val=5;
else if(v(4,3,2,x,y)==1 && sn<8) val=3;
#endif

/* for odd */
#if Odd==3
else if(v(2,2,0,x,y)==1 && sn>0) val=0;
else if(v(2,2,1,x,y)==1 && sn>0) val=2;
else if(v(2,2,2,x,y)==1 && sn>0) val=4;
#endif

else val=-1;

x+=sn*(RESO-1);

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_;

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 cag_r(void)
{
int i,j,ds,dx,n,sn1,sn2,jmax,dsn;
int flag_[CPMAX],flag_pp[CPMAX],acolor[3*2];
int nx[CPMAX],ny[CPMAX],nx_[CPMAX],ny_[CPMAX],nax[3*2],nay[3*2];
int Nx,Ny,x[8],y[8],x_[8],y_[8];
int cp,ssize,posflag,pos,count;
int ca,c1,c2,c3,c4,c5,c7,cflag,cc1,cc2;
int nxp,nxm,nyp,nym;
int jmp[6+6],sntmp1,sntmp2,snold;

```

```

ssize=sizeof(ss);
cp=CPMAX;
n=RES0;
ds=n-1;

acolor[0]=9;acolor[1]=10;acolor[2]=11;acolor[3]=12;acolor[4]=13;acolor[5]=14;

for(i=0;i<CPMAX;i++){
rcount[i]=0;
flag_[i]=1;
fp_mem[i]=0;
}

/*999*/
ca=15;

if(Odd>0){
nax[0]=1+ds      ;nay[0]=2+ds; /* 2-2(sn=0) */
nax[1]=1+ds      ;nay[1]=n-2+ds;
nax[2]=n-3+ds;nay[2]=n-2+ds;

if(Odd==1) dx=0;
else dx=8*(n-1);
/* for odd */
nax[3]=2+3*ds+dx      ;nay[3]=1+ds+3*ds; /* 4-3(sn=0, 8) */
nax[4]=n-2+3*ds+dx;nay[4]=n-3+ds+3*ds;
nax[5]=n-2+3*ds+dx;nay[5]=1+ds+3*ds;
}
else{
dx=/*0*//*4*(n-1)*/0;
nax[0]=1+ds+dx      ;nay[0]=2+ds; /* 2-2(sn=0) */
nax[1]=1+ds+dx      ;nay[1]=n-2+ds;
nax[2]=n-3+ds+dx;nay[2]=n-2+ds;

dx=4*(n-1);
/* for even */
nax[3]=1+ds+dx      ;nay[3]=2+ds; /* 2-2(sn=4) */
nax[4]=1+ds+dx      ;nay[4]=n-2+ds;
nax[5]=n-3+ds+dx;nay[5]=n-2+ds;
}

if(Odd>0){
/* for odd */
if(Odd==1) {sn1=0;sn2=0;}
else {sn1=0;sn2=8;}
}

```

```

else{
/* for even */
sn1=/*0*//*4*/0;sn2=4;
}

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];
putpixel_(nx[i],ny[i],acolor[i]);
}/**if(flag_[i])**/

i++;if(i==CPMAX) 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(combination==0){
}/**if(combination)**/
else{
/* CCW */
if(drn==4) {/* 217 */
if(Odd>0 && CPMAX==6 && i>=3){
/* 4-3 */
if(i%3==0) {nax[i]++;nay[i]++;}
else if(i%3==1) {nay[i]--;}
else if(i%3==2) {nax[i]--;}
}
else{
/* 2-2 */
if(i%6==0) {nay[i]++;}
else if(i%6==1) {nax[i]++;}
else if(i%6==2) {nax[i]--;nay[i]--;}

/*else if(i%6==3) {nay[i]++;}
else if(i%6==4) {nax[i]++;}
else if(i%6==5) {nax[i]--;nay[i]--;}*/
else if(i%6==3) {nax[i]++;nay[i]++;}

```

```

else if(i%6==4) {nay[i]--;}
else if(i%6==5) {nax[i]--;}
}
}
}/**else(combination)**/

nx[i]=nax[i];ny[i]=nay[i];

putpixel_(nx[i],ny[i],acolor[i]);
}/**if(flag_[i])**/

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

if(GRPH==1 && cnt==0) use_subroop();
cnt++;
/***** while(cp) -> *****/

while(cp){
kbhit_();
if(refill==0) break;
/*if(cnt==103 && rcount[0]>=66){
printf(" %ld %ld %ld\n",rcount[0],rcount[1],rcount[2]);
use_subroop();
}*/

algo=random_(2);

i=0;
while(1){

if(flag_[i]){ /* CP_? */
if(Odd!=1) {if(i<=2) sn=sn1;else sn=sn2;}
else sn=sn1;
ig=i;

nxp=nx[i]+1;nyp=ny[i]+1;nxm=nx[i]-1;nym=ny[i]-1;
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_;
c5=getpixel_(nx[i],ny[i],nxp,nyp);

```



```

x[5]=X;y[5]=Y;x_[5]=X_;y_[5]=Y_;jmp[5]=sn_;
c7=getpixel_(nx[i],ny[i],nxm,nym);
x[7]=X;y[7]=Y;x_[7]=X_;y_[7]=Y_;jmp[7]=sn_;

Nx=nx[i];Ny=ny[i];

if(sn==0) dsn=4;
else if(sn==4 && Ny>3*(n-1)) dsn=-4;
else if(sn==4 && Ny<3*(n-1)) dsn=4;
else if(sn==8) dsn=-4;

posflag=check_v(Nx,Ny);
cflag=0;

if(posflag==-1){
if((c1==ca)||(c2==ca)||(c3==ca)||(c4==ca)||(c5==ca)||(c7==ca)) cflag=1;
else cflag=0;
}
else if(posflag==0){
if((c1==ca)||(c2==ca)||(c3==ca)||(c4==ca)||(c5==ca)||(c7==ca)) cflag=1;
else{
j=6;rot(2,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(2,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(2,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(2,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

/*cflag=0;
for(j=6;j<=9;j++) {if(pixel[enX[j]][enY[j]]==ca) {cflag=1;break;}}*/
}
}
else if(posflag==1){
if((c1==ca)||(c2==ca)||(c3==ca)||(c4==ca)||(c5==ca)||(c7==ca)) cflag=1;
else{
j=6;rot(1,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(1,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(1,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(1,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;

```

```

set_vals(j);

/*cflag=0;
for(j=6;j<=9;j++) {if(pixel[enX[j]][enY[j]]==ca) {cflag=1;break;}}*/
}
}
else if(posflag==2){
if((c1==ca)|| (c2==ca)|| (c3==ca)|| (c4==ca)|| (c5==ca)|| (c7==ca)) cflag=1;
else{
j=6;rot(0,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(0,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(0,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(0,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

/*cflag=0;
for(j=6;j<=9;j++) {if(pixel[enX[j]][enY[j]]==ca) {cflag=1;break;}}*/
}
}
else if(posflag==3){
if((c1==ca)|| (c2==ca)|| (c3==ca)|| (c4==ca)|| (c5==ca)|| (c7==ca)) cflag=1;
else{
j=6;rot(5,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(5,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(5,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(5,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

/*cflag=0;
for(j=6;j<=9;j++) {if(pixel[enX[j]][enY[j]]==ca) {cflag=1;break;}}*/
}
}
else if(posflag==4){
if((c1==ca)|| (c2==ca)|| (c3==ca)|| (c4==ca)|| (c5==ca)|| (c7==ca)) cflag=1;
else{
j=6;rot(4,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(4,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(4,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;

```

```

set_vals(j);
j=9;rot(4,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

/*cflag=0;
for(j=6;j<=9;j++) {if(pixel[enX[j]][enY[j]]==ca) {cflag=1;break;}}*/
}
}
else if(posflag==5){
if((c1==ca)||(c2==ca)||(c3==ca)||(c4==ca)||(c5==ca)||(c7==ca)) cflag=1;
else{
j=6;rot(3,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(3,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(3,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(3,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

/*cflag=0;
for(j=6;j<=9;j++) {if(pixel[enX[j]][enY[j]]==ca) {cflag=1;break;}}*/
}
}

if(posflag!=-1 && cflag==0){
snold=sn;
sn=sn_;

for(j=6;j<=9;j++){
cc1=/*getpixel_(X_,Y_,enX[j],enY[j]);*/pixel[enX[j]][enY[j]];
if(cc1==ca) {cflag=1;break;}
}

sn=snold;
}

if(cflag){
s.xx=nx[i];s.yy=ny[i];s.xx_=nx_[i];s.yy_=ny_[i];s.sn=sn;fwrite_mem(i);

std_x=nx[i];std_y=ny[i]; /* S */
last_x=nx_[i];last_y=ny_[i];
/*nx_[i]=nx[i];ny_[i]=ny[i];*/ /* new b */

```

```

/* for out */
/*if(sn==0) dsn=4;
else if(sn==4 && Ny>3*(n-1)) dsn=-4;
else if(sn==4 && Ny<3*(n-1)) dsn=4;
else if(sn==8) dsn=-4;*/

posflag=check_v(Nx,Ny);

if(posflag==-1){
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];
}

jmax=j;
}
else if(posflag==0){
/* 0v0 */
j=0; /* ca5 */
enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=1; /* ca1 */
enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=2; /* ca4 */

```

```

    enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=3; /* ca7 */
    enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=4; /* ca3 */
    enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=5; /* ca2 */
    enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];

        /* out */
j=6;rot(2,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(2,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(2,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(2,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

jmax=9;
}
else if(posflag==1){
/* 1v0 */
j=0; /* ca1 */
    enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=1; /* ca4 */
    enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=2; /* ca7 */
    enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=3; /* ca3 */
    enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=4; /* ca2 */
    enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=5; /* ca5 */
    enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];

        /* out */
j=6;rot(1,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(1,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(1,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(1,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

jmax=9;

```

```

}
else if(posflag==2){
/* 2v1 */
j=0; /* ca4 */
enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=1; /* ca7 */
enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=2; /* ca3 */
enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=3; /* ca2 */
enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=4; /* ca5 */
enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=5; /* ca1 */
enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];

/* out */
j=6;rot(0,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(0,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(0,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(0,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

jmax=9;
}
else if(posflag==3){
/* 3v2 */
j=0; /* ca7 */
enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=1; /* ca3 */
enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=2; /* ca2 */
enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=3; /* ca5 */
enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=4; /* ca1 */
enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=5; /* ca4 */
enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];

/* out */
j=6;rot(5,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

```

```

j=7;rot(5,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(5,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(5,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

jmax=9;
}
else if(posflag==4){
/* 4v2 */
j=0; /* ca3 */
enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];
j=1; /* ca2 */
enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=2; /* ca5 */
enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=3; /* ca1 */
enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=4; /* ca4 */
enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];
j=5; /* ca7 */
enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];

/* out */
j=6;rot(4,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(4,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(4,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(4,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

jmax=9;
}
else if(posflag==5){
/* 5v1 */
j=0; /* ca2 */
enX[j]=x[2];enY[j]=y[2];enX_[j]=x_[2];enY_[j]=y_[2];enSN[j]=jmp[2];
j=1; /* ca5 */
enX[j]=x[5];enY[j]=y[5];enX_[j]=x_[5];enY_[j]=y_[5];enSN[j]=jmp[5];
j=2; /* ca1 */
enX[j]=x[1];enY[j]=y[1];enX_[j]=x_[1];enY_[j]=y_[1];enSN[j]=jmp[1];
j=3; /* ca4 */
enX[j]=x[4];enY[j]=y[4];enX_[j]=x_[4];enY_[j]=y_[4];enSN[j]=jmp[4];

```

```

j=4; /* ca7 */
    enX[j]=x[7];enY[j]=y[7];enX_[j]=x_[7];enY_[j]=y_[7];enSN[j]=jmp[7];
j=5; /* ca3 */
    enX[j]=x[3];enY[j]=y[3];enX_[j]=x_[3];enY_[j]=y_[3];enSN[j]=jmp[3];

                /* out */
j=6;rot(3,1);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=7;rot(3,2);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=8;rot(3,3);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);
j=9;rot(3,4);X=Nx+tmp0;Y=Ny+tmp1;X_=Nx;Y_=Ny;X+=dsn*(n-1);X_+=dsn*(n-1);sn_=sn+dsn;
set_vals(j);

jmax=9;
}

for(j=0;j<=jmax;j++){
if(enX[j]==nx_[i] && enY[j]==ny_[i]) {pos=j;break;}
}

if(algo==0){
/* CW */
count=0;
for(j=pos;;){
cc1=pixel[fen("X",j,jmax)][fen("Y",j,jmax)];
cc2=pixel[fen("X",j-1,jmax)][fen("Y",j-1,jmax)];

/*if(pixel[fen("X",j,jmax)][fen("Y",j,jmax)]!=ca &&
    pixel[fen("X",j-1,jmax)][fen("Y",j-1,jmax)]==ca){*/
if(cc1!=ca && cc2==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<=2) 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;;){
cc1=pixel[fen("X",j,jmax)][fen("Y",j,jmax)];
cc2=pixel[fen("X",j+1,jmax)][fen("Y",j+1,jmax)];

/*if(pixel[fen("X",j,jmax)][fen("Y",j,jmax)]!=ca &&
    pixel[fen("X",j+1,jmax)][fen("Y",j+1,jmax)]==ca){*/
if(cc1!=ca && cc2==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<=2) 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()**/

/*printf(" %ld i:%d sn:%d posflag:%d\n",rcount[i],i,sn,posflag);
if(i==2 && posflag==3){
printf(" al:%d pos:%d\n",algo,pos);
for(j=0;j<=9;j++)
printf(" j:%d X:%d Y:%d SN_:%d val:%d\n",
        j,enX[j],enY[j],enSN[j],pixel[enX[j]][enY[j]]);
}*/

if(1){
if(i<CPHALF-1){
sn=sntmp1;
}
else if(i==CPHALF-1){
sn=sntmp1;
sn1=sntmp1;
}
else if(i>CPHALF-1 && i<CPMAX-1){
sn=sntmp2;
}
else if(i==CPMAX-1){
sn=sntmp2;
sn2=sntmp2;
}
putpixel_(nx[i],ny[i],acolor[i]);
}

```

```
flag_pp[i]=1;
}/**if(c1,c2,c3,c4)**/
else{
if(ftell_mem(i)==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(i==CPMAX) break;
}/**while(1)**/

if(0){
}
}/**while(cp)**/

/*putpixel(3*(n-1),4*(n-1),0);
putpixel((3+4)*(n-1),4*(n-1),0);*/

return 0;
}/** cag_r **/
```