Cellular Automaton Graphics(3)

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. Octahedron

Figure 1 is developments of octahedron. (1) is a normal development. The regular triangle at the far left in (1) is moved to the far right in (2) in order that we can deal with it easily. The number is the number of the vertex, side and the following are combinations of numbers of vertexes, two sides in correspondence.

1, 3；2, 6；8, 10；5, 9；11(1-2), 15(3-6)；12(4-3), 17(4-1)；13(7-8), 18(7-10)；14(9-10), 20(5-8)；16(6-9), 19(2-5)

"1, 3；2, 6；11(1-2), 15(3-6)；12(4-3), 17(4-1)" are inversion on $y = x$ and "8, 10；5, 9；13(7-8), 18(7-10)；14(9-10), 20(5-8)" are inversion on $y = x + (n-1)$. "16(6-9), 19(2-5)" are parallel movement in which Δx is $\pm 2(n-1)$ and Δy is $\mp 2(n-1)$.

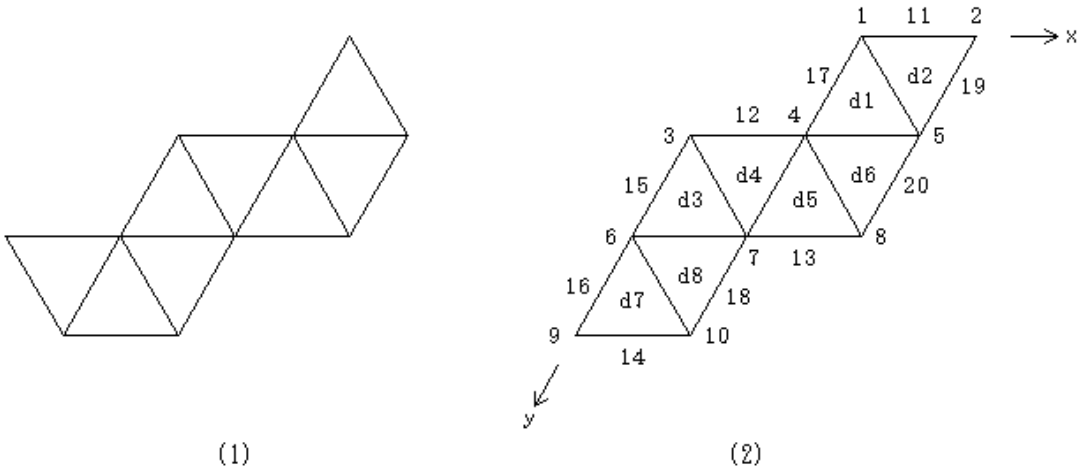
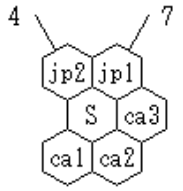


Figure 1

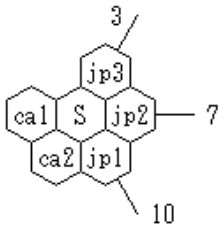
2. Neighborhood view

Figure 2 is neighborhood views on vertex, side. In octahedron, there are four regular triangles around a vertex. Therefore, 0〜3 regular triangles are added to each vertex in Figure 1. For the reason which was mentioned the last, the adding is not required substantially in vertex 5, 6.



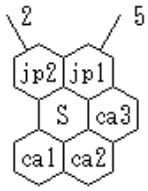
S:1(3)
jp1:(1,n)
jp2:(n-1,1) <=> ca1

vertex 1



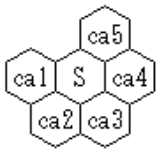
S:2(6)
jp1:(1,2n-1)
jp2:(1,2n-2)
jp3:(0,2n-3) <=> ca1

vertex 2



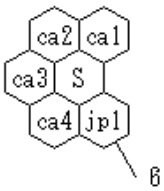
S:3(1)
jp1:(n,1)
jp2:(n,0) <=> ca1

vertex 3



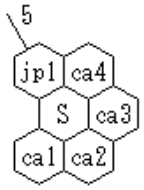
S:4

vertex 4



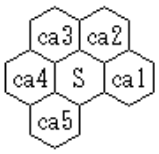
S:5(9)
jp1:(0,3n-4) <=> ca1

vertex 5



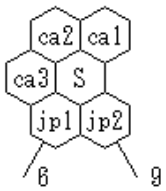
S:6(2)
jp1:(2(n-1),1) <=> ca1

vertex 6



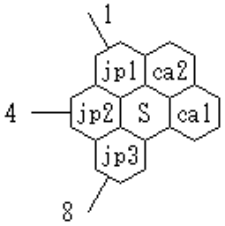
S:7

vertex 7



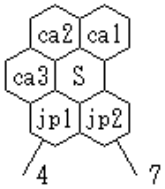
S:8(10)
jp1:(n-2,3n-4)
jp2:(n-2,3(n-1)) <=> ca1

vertex 8



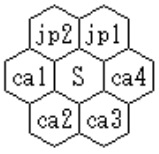
S:9(5)
jp1:(2n-3,n-2)
jp2:(2n-3,n-1)
jp3:(2(n-1),n) <=> ca1

vertex 9



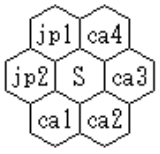
S:10(8)
jp1:(2n-3,2n-3)
jp2:(n-1,3n-4) <=> ca1

vertex 10



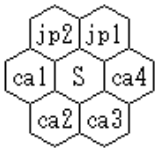
S:(x,0)
jp1:(1,x+1)
jp2:(1,x)

side 11



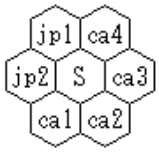
S:(0,y)
jp1:(y,1)
jp2:(y+1,1)

side 15



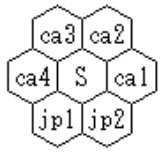
S:(x,n-1)
jp1:(n,x+1)
jp2:(n,x)

side 12

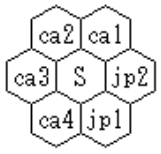


S:(n-1,y)
jp1:(y,n)
jp2:(y+1,n)

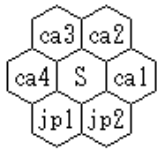
side 17



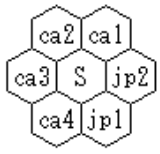
S:(x,2(n-1))
jp1:(n-2,x+n-2)
jp2:(n-2,x+(n-1))
side 13



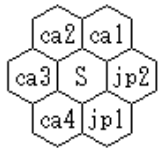
S:(n-1,y)
jp1:(y-(n-1),2n-3)
jp2:(y-n,2n-3)
side 18



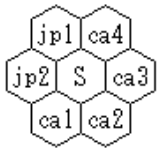
S:(x,3(n-1))
jp1:(2n-3,x+n-2)
jp2:(2n-3,x+(n-1))
side 14



S:(2(n-1),y)
jp1:(y-(n-1),3n-4)
jp2:(y-n,3n-4)
side 20



S:(2(n-1),y)
jp1:(1,y+1+2(n-1))
jp2:(1,y+2(n-1))
side 19



S:(0,y)
jp1:(2n-3,y-1-2(n-1))
jp2:(2n-3,y-2(n-1))
side 16

Figure 2

3. Seed point

We distribute seed points to an octahedron which is shown in Figure 1-(2).

(1) SP(3 × 1):3(d1)

d1 is a number of a regular triangle to which seed points are distributed. If three seed points are distributed to a regular triangle, three symmetric graphics are drawn in another regular triangle too. We call the two regular triangles convergence surfaces.

(2) SP(3 × 2):3(d1)+3(d8):d?-c

When six seed points are distributed equally to two regular triangles, two convergence surfaces are chosen. d?-c represents that the two regular triangles are convergence surfaces each other.

(3) SP(3 × 4):3(d1)+3(d3)+3(d5)+3(d7):d?-v, CP(all)

d?-v represents that the regular triangles connect at vertex. CP represents vertexes to close.

(4) SP(3 × 8):3(d1)+3(d2)+,,,+3(d7)+3(d8):d?-s, CP(all)

d?-s represents that the regular triangles connect at side.

(5) SP(1 × 2):1(d3)+1(d6):d?-c

(6) SP(1 × 2):1(d3)+1(d4):d?-s

(7) SP(1 × 2):1(d1)+1(d5):d?-v, CP(2):4, 6;2

(8) SP(1 × 4):1(d1)+1(d4)+1(d5)+1(d6):d?-sv, CP(2):4, 6;2

d?-sv represents that the regular triangles connect at side and they are around a common vertex.

(9) SP(1 × 3):1(d2)+1(d4)+1(d6):d?-vr
d?-vr represents that the regular triangles connect at vertex and they are in a ring.

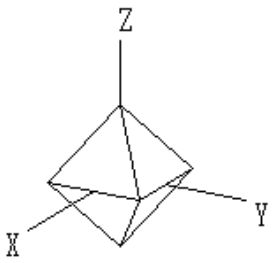
The coordinates of seed points and next points(painting number 2) are fixed in consideration of symmetry. Vector which is fixed by a seed point and its painting number 2 is called 1st vector. If 1st vector of one painting point is given, seed points and painting number 2 of the other painting points are fixed by a roatation of 1st vector around a symmetric axis. In Figure 3, we assume that the vertexes 4, 6;2 are on Z axis. For example, the symmetric axis of (7) SP(1 × 2) is Z axis and the symmetric axes of (6) SP(1 × 2) and (5) SP(1 × 2) are the other two axes.

4. Convergence surface

In convergence surface, more than one painting points draws symmetric graphics and comes closest. Expanding the definition, we call not only one regular triangle but also a group of regular triangles in which symmetric graphics are drawn by more than one painting points convergence surface. We assume that seed points of (7) SP(1 × 2) and (8) SP(1 × 4) are in the upper half in Figure 3. Because symmetric graphics are drawn in the upper and lower half, the convergence surfaces are the upper and lower half. Therefore, in SP(1 × 2 × 2):1(d1)+1(d5)+1(d3)+1(d7):(d?-v) × 2, CP(2):4, 6;2 and SP(1 × 4 × 2):1(d1)+1(d4)+1(d5)+1(d6)+1(d2)+1(d3)+1(d7)+1(d8):(d?-sv) × 2, CP(all) in which seed points are distributed to both convergence surfaces, too, symmetric graphics are drawn in both convergence surfaces.

In (9) SP(1 × 3):d?-vr, too, convergence surfaces exist. It has a shape like claw clutch. In SP(1 × 3 × 2):1(d2)+1(d4)+1(d6)+1(d3)+1(d5)+1(d7):(d?-vr) × 2 in which seed points are distributed to both convergence surfaces, too, symmetric graphics are drawn in both convergence surfaces.

Convergence surface is a group of regular triangles which was created in order to search all seed points. There is a limit in seed points which is got on general geometric symmetry. We can squeeze seed points by thinking various convergence surfaces searching symmetric graphics on a polyhedron actually.



#define _6dRow 1

Figure 3

5. Modification of region by less than tetrahedron

If four regular triangles of octahedron are replaced with less than tetrahedron, a polyhedron like Figure 4 is got. We show how to make this polyhedron in Figure 5. First, there are four regular triangles which is marked with R. If the Rs are replaced with the replacement figure one by one, the number of regular triangles becomes 16 finally. In the figure, a correspondence of two line segments is represented by their color and arrows. If the two line segments is not connected, their color is unique and a color of the two line segments is changed flexibly. If (8) is done inversion and rotation as a whole, it corresponds with (7).

Vertexes are classified like the following:

・vertexes around which 3 regular triangles are：2, 4, 12, 14
・vertexes around which 6 regular triangles are：7, 8, 9, 6;10, 1;3;5, 11;13;15

Transformations of side are all inversion.

・16⇔13, 20⇔21：inversion on $x = 2(n-1)$
・18⇔19, 22⇔23：inversion on $x = 6(n-1)$
・24⇔26, 25⇔27：inversion on $x = 4(n-1)$

We adopt SP(1 × 2):1(d1)+1(d12):d?-v, CP(2):7, 9 for example as seed point. Because the polyhedron of Figure 4 looks like a tetrahedron as a whole, we can reason that SP(1 × 2):1(d1)+1(d2):d?-s is possible in tetrahedron.

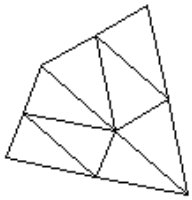


Figure 4

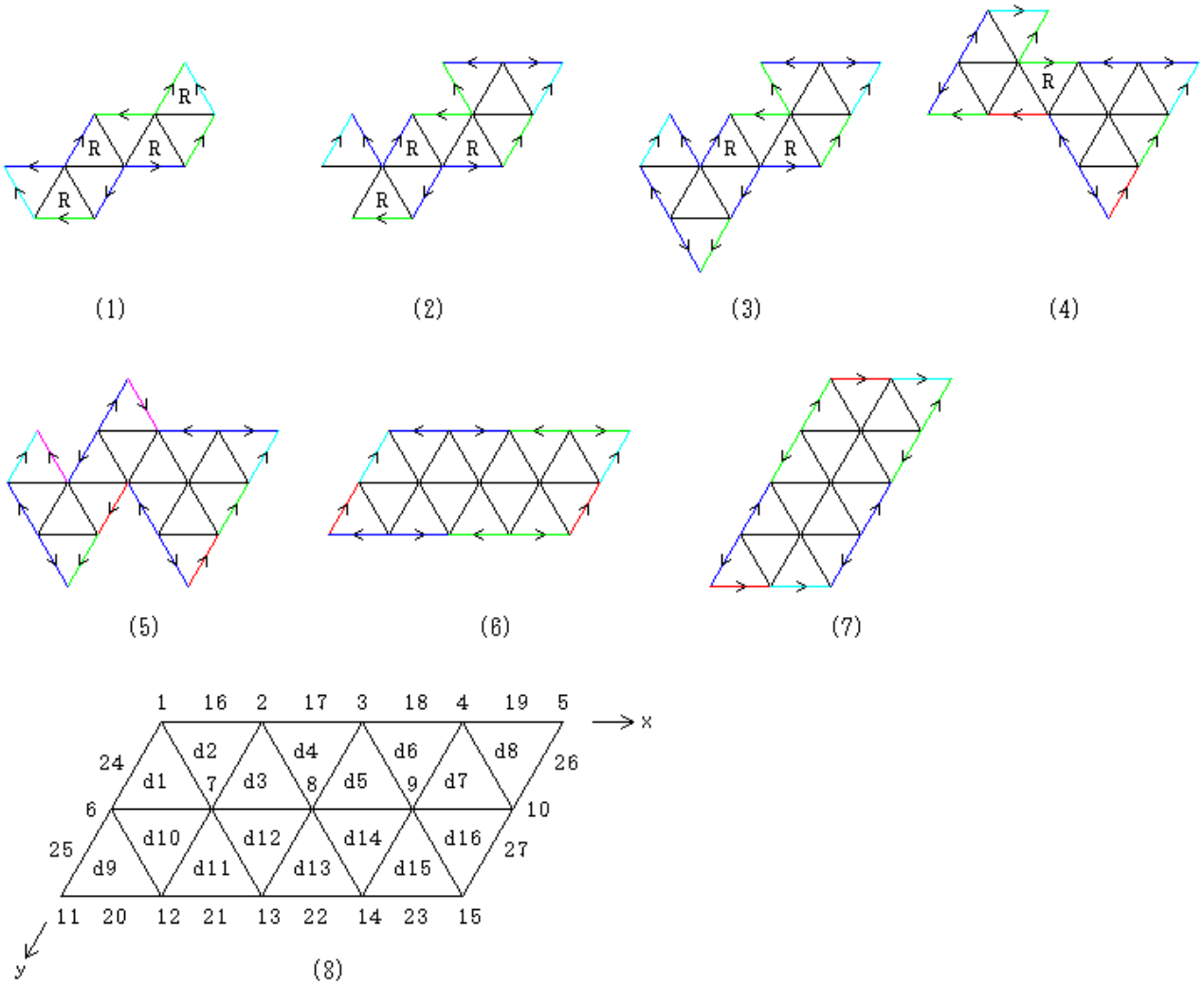


Figure 5

6. Modification of region by less than octahedron

If some octahedrons are connected, polyhedrons like Figure 6 are got. Figure 6-(1) is got if one regular triangle of octahedron is replaced with the figure(less than octahedron) of which development is Figure 7. We show how to make this polyhedron in Figure 8. If the regular triangle R is replaced with the replacement figure, the number of regular triangles becomes 14.

・vertexes around which 4 regular triangles are : 2, 4, 11, 13, 1;3;5, 10;12;14
・vertexes around which 6 regular triangles are : 7, 8, 6;9

Transformations of side are all inversion.

・23⇔25, 24⇔26 : inversion on $x = 3(n-1)/2$
・15⇔16 : inversion on $y = x$
・21⇔22 : inversion on $y = x + (n-1)$
・17⇔18 : inversion in terms of orthogonal coordinates on $x = 2(n-1)$
・19⇔20 : inversion in terms of orthogonal coordinates on $x = n-1$

We adopt SP(3×1):3(d1) for example as seed point. The minimum convergence surface in this case is d1, d14 and if we regard Figure 6-(1) as a tunnel, they are its two mouths.

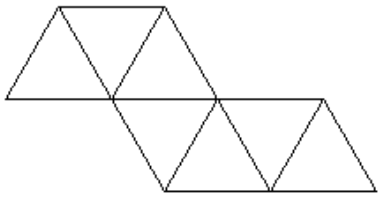
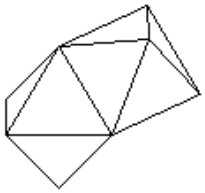
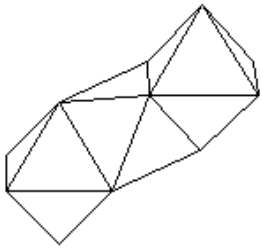


Figure 6



#define _6dRow 2

(1)



#define _6dRow 3

(2)

Figure 7

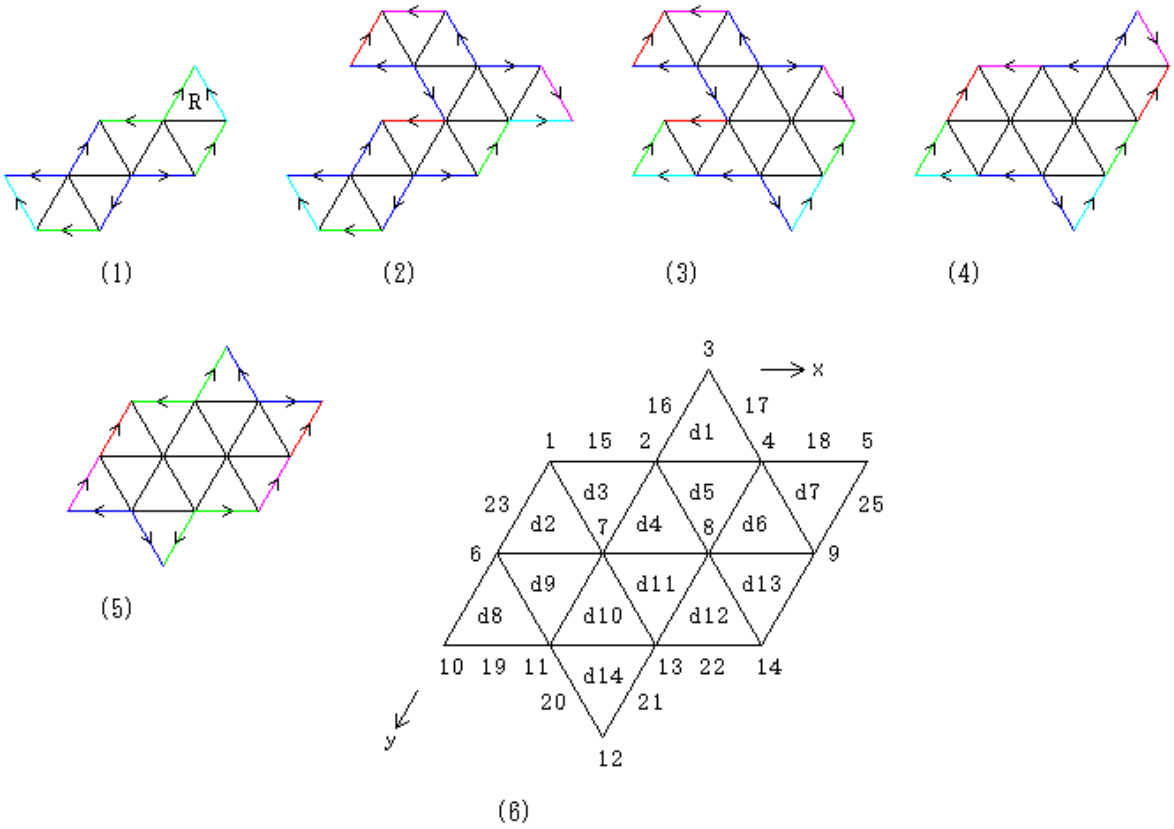


Figure 8

7. Assignment

・Try making neighborhood views on vertexes and sides of the figure of Figure 4, Figure 7.

・Try getting convergence surfaces in the direction of the tunnel of the figure of Figure 7. We call octahedron, the figure of Figure 7-(2) odd tunnel and the figure of Figure 7-(1) even tunnel.

8. Concrete example

The detailed data of the above seed points are as follows:

(1) SP(3×1):3(d1)

・$n = 12$

・coordinates of painting number 1：painting point a:$(n-1+2,\ 4)$、painting point b:$(n-1+n-5,\ n-3)$、painting point c:$(n-1+2,\ n-3)$

・coordinates of painting number 2：painting point a:$\Delta y = 1$、painting point b:$\Delta x = -1; \Delta y = -1$、painting point c:$\Delta x = 1$

(2) SP(3×2):3(d1)+3(d8):d?-c

・$n = 12$

・coordinates of painting number 1：painting point a_0:$(n-1+2,\ 4)$、painting point b_0:$(n-1+n-5,\ n-3)$、painting point c_0:$(n-1+2,\ n-3)$

・coordinates of painting number 1：painting point a_1':$(4,\ 2(n-1)+2)$、painting point b_1':$(n-3,\ 2(n-1)+2)$、painting point c_1':$(n-3,\ 3(n-1)-4)$

・coordinates of painting number 2：painting point a_0:$\Delta y = 1$、painting point b_0:$\Delta x = -1; \Delta y = -1$、painting point c_0:$\Delta x = 1$

・coordinates of painting number 2：painting point a_1':$\Delta x = 1; \Delta y = 1$、painting point b_1':$\Delta x = -1$、

painting point c'_1:$\Delta y = -1$

(3) SP(3×4):3(d1)+3(d3)+3(d5)+3(d7):d?-v, CP(all)
・$n = 12$
・coordinates of painting number 1：painting point a_0:$(n-1+2,\ 4)$、painting point b_0:$(n-1+n-5,\ n-3)$、painting point c_0:$(n-1+2,\ n-3)$；painting point a_i、painting point b_i、painting point $c_i (i > 0)$:parallel movement of d1(a_0、b_0、c_0)
・coordinates of painting number 2：painting point a_i:$\Delta y = 1$、painting point b_i:$\Delta x = -1; \Delta y = -1$、painting point c_i:$\Delta x = 1$

(4) SP(3×8):3(d1)+3(d2)+,,,+3(d7)+3(d8):d?-s, CP(all)
・$n = 12$
・coordinates of painting number 1：painting point a_0:$(n-1+2,\ 4)$、painting point b_0:$(n-1+n-5,\ n-3)$、painting point c_0:$(n-1+2,\ n-3)$；painting point a_i、painting point b_i、painting point $c_i (0 < i < 4)$:parallel movement of d1(a_0、b_0、c_0)
・coordinates of painting number 1：painting point a'_4:$(n-1+4,\ 2)$、painting point b'_4:$(n-1+n-3,\ 2)$、painting point c'_4:$(n-1+n-3,\ n-5)$；painting point a'_i、painting point b'_i、painting point $c'_i (i > 4)$:parallel movement of d2(a'_4、b'_4、c'_4)
・coordinates of painting number 2：painting point a_i:$\Delta y = 1$、painting point b_i:$\Delta x = -1; \Delta y = -1$、painting point c_i:$\Delta x = 1 \quad (i < 4)$
・coordinates of painting number 2：painting point a'_i:$\Delta x = 1; \Delta y = 1$、painting point b'_i:$\Delta x = -1$、painting point c'_i:$\Delta y = -1 \quad (i > 3)$

(5) SP(1×2):1(d3)+1(d6):d?-c
・$n = 12$
・coordinates of painting number 1：painting point c_0:$(2,\ n-1+n-3)$
・coordinates of painting number 1：painting point a'_1:$(n-1+4,\ n-1+2)$
・coordinates of painting number 2：painting point c_0:$\Delta y = -1$
・coordinates of painting number 2：painting point a'_1:$\Delta x = 1$

(6) SP(1×2):1(d3)+1(d4):d?-s
・$n = 12$
・coordinates of painting number 1：painting point c_0:$(2,\ n-1+n-3)$
・coordinates of painting number 1：painting point b'_1:$(n-3,\ n-1+2)$
・coordinates of painting number 2：painting point c_0:$\Delta y = -1$
・coordinates of painting number 2：painting point b'_1:$\Delta y = 1$

(7) SP(1×2):1(d1)+1(d5):d?-v, CP(2):4, 6;2
・$n = 12$
・coordinates of painting number 1：painting point c_0:$(n-1+2,\ n-3)$
・coordinates of painting number 1：painting point a_1:$(n-1+2,\ n-1+4)$
・coordinates of painting number 2：painting point c_0:$\Delta x = 1$
・coordinates of painting number 2：painting point a_1:$\Delta y = 1$

(8) SP(1×4):1(d1)+1(d4)+1(d5)+1(d6):d?-sv, CP(2):4, 6;2
・$n = 12$

・coordinates of painting number 1：painting point c_0:$(n-1+2,\ n-3)$
・coordinates of painting number 1：painting point b'_1:$(n-3,\ n-1+2)$
・coordinates of painting number 1：painting point a_2:$(n-1+2,\ n-1+4)$
・coordinates of painting number 1：painting point a'_3:$(n-1+4,\ n-1+2)$
・coordinates of painting number 2：painting point c_0:$\Delta x=1$
・coordinates of painting number 2：painting point b'_1:$\Delta x=-1$
・coordinates of painting number 2：painting point a_2:$\Delta y=1$
・coordinates of painting number 2：painting point a'_3:$\Delta x=1;\Delta y=1$

(9) SP(1 × 3):1(d2)+1(d4)+1(d6):d?-vr
・$n=12$
・coordinates of painting number 1：painting point a'_0:$(n-1+4,\ 2)$
・coordinates of painting number 1：painting point b'_1:$(n-1-2,\ n-1+2)$
・coordinates of painting number 1：painting point b'_2:$(2(n-1)-2,\ n-1+2)$
・coordinates of painting number 2：painting point a'_0:$\Delta x=1;\Delta y=1$
・coordinates of painting number 2：painting point b'_1:$\Delta x=-1$
・coordinates of painting number 2：painting point b'_2:$\Delta x=-1$

Figure 9 is a symmetrical graphics by Figure 2 and the following are its data.

・SP(3 × 1):3(d1)
・$n=12$
・coordinates of painting number 1：painting point a:$(n-1+2,\ 4)$、painting point b:$(n-1+n-5,\ n-3)$、painting point c:$(n-1+2,\ n-3)$
・coordinates of painting number 2：painting point a:$\Delta y=1$、painting point b:$\Delta x=-1;\Delta y=-1$、painting point c:$\Delta x=1$

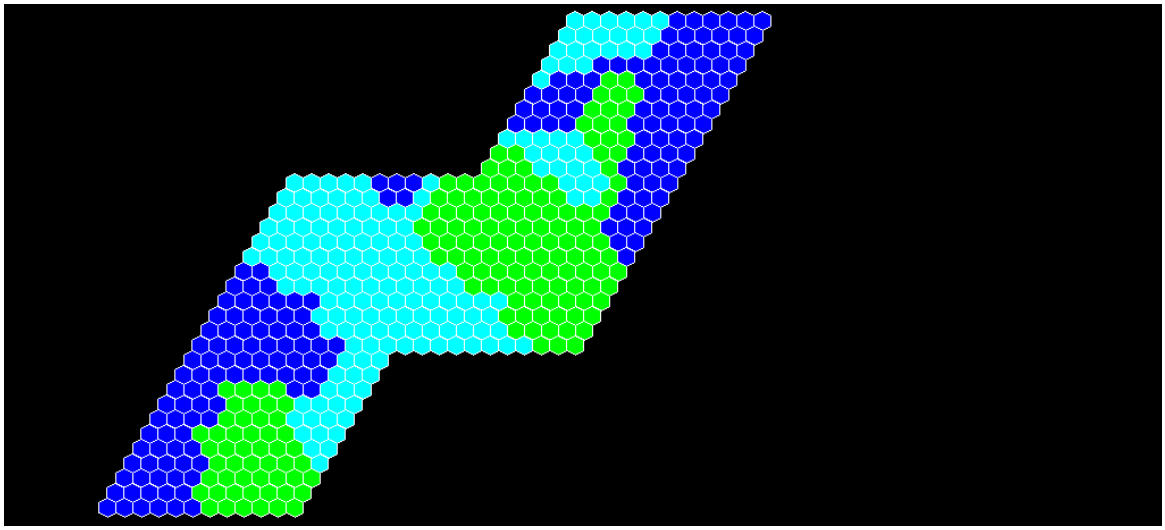


Figure 9

Figure 10 is a symmetrical graphics by Figure 5 and the following are its data.

・$n=12$

・SP(1×2):1(d1)+1(d12):d?-v, CP(2):7, 9
・coordinates of painting number 1：painting point b_0:$(n-5,\ n-3)$
・coordinates of painting number 1：painting point a_1':$(n-1+4,\ n-1+2)$
・coordinates of painting number 2：painting point b_0:$\Delta x=-1$
・coordinates of painting number 2：painting point a_1':$\Delta x=1$

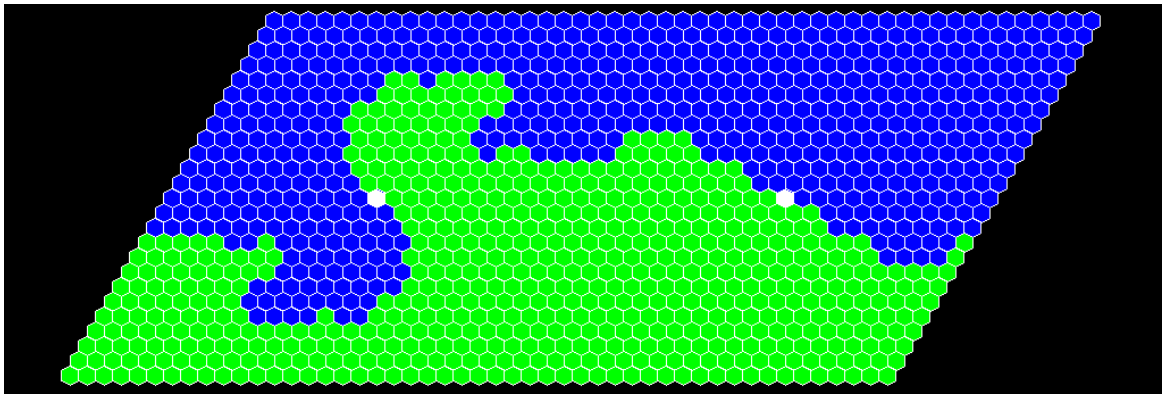


Figure 10

Figure 11 is a symmetrical graphics by Figure 8 and the following are its data.

・SP(3×1):3(d1)
・$n=12$
・coordinates of painting number 1：painting point a:$(n-1+2,\ 4)$、painting point b:$(n-1+n-5,\ n-3)$、painting point c:$(n-1+2,\ n-3)$
・coordinates of painting number 2：painting point a:$\Delta y=1$、painting point b:$\Delta x=-1;\Delta y=-1$、painting point c:$\Delta x=1$

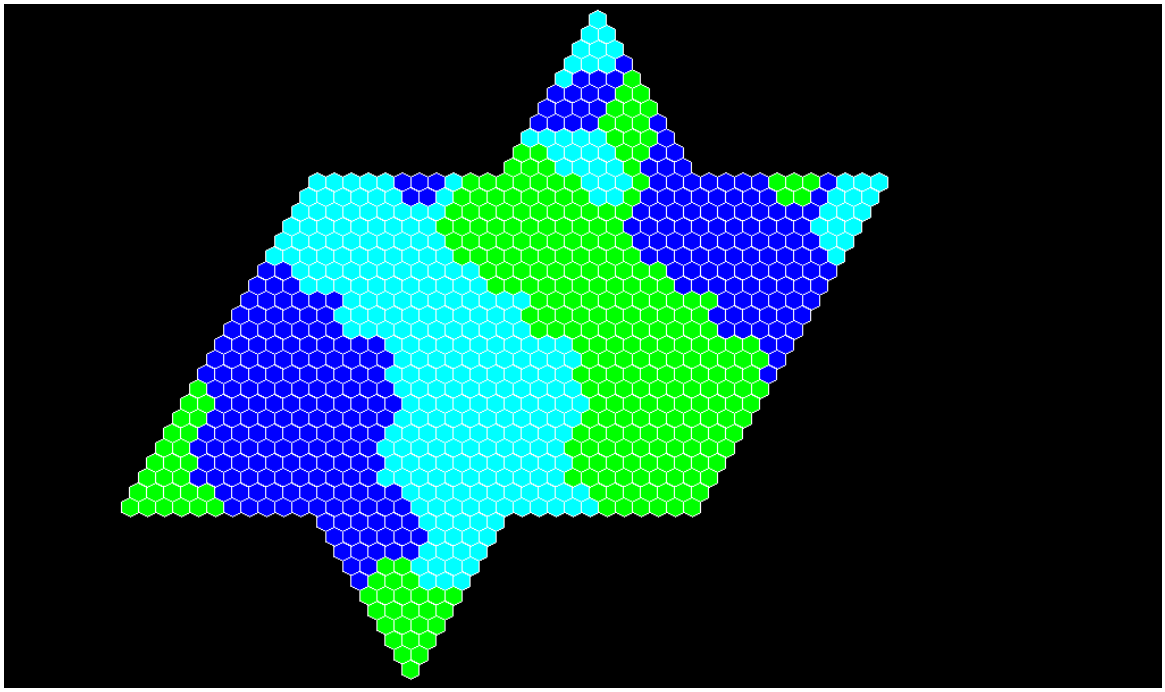


Figure 11

****

セルラーオートマトングラフィクス (3)

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 8 面体

　図 1 は正八面体の展開図です。(1) は普通の展開図です。(1) においてプログラム上扱いやすいように左端の正三角形を移動させたものが (2) です。数字は頂点、辺の番号であり、以下は対応する頂点、辺の番号の組合せです。

1, 3；2, 6；8, 10；5, 9；11(1-2), 15(3-6)；12(4-3), 17(4-1)；13(7-8), 18(7-10)；14(9-10), 20(5-8)；16(6-9), 19(2-5)

”1, 3；2, 6；11(1-2), 15(3-6)；12(4-3), 17(4-1)”は $y = x$ に関する反転であり、”8, 10；5, 9；13(7-8), 18(7-10)；14(9-10), 20(5-8)”は $y = x + (n - 1)$ に関する反転です。”16(6-9), 19(2-5)”は x 方向が $\pm 2(n - 1)$、y 方向が $\mp 2(n - 1)$ の平行移動です。

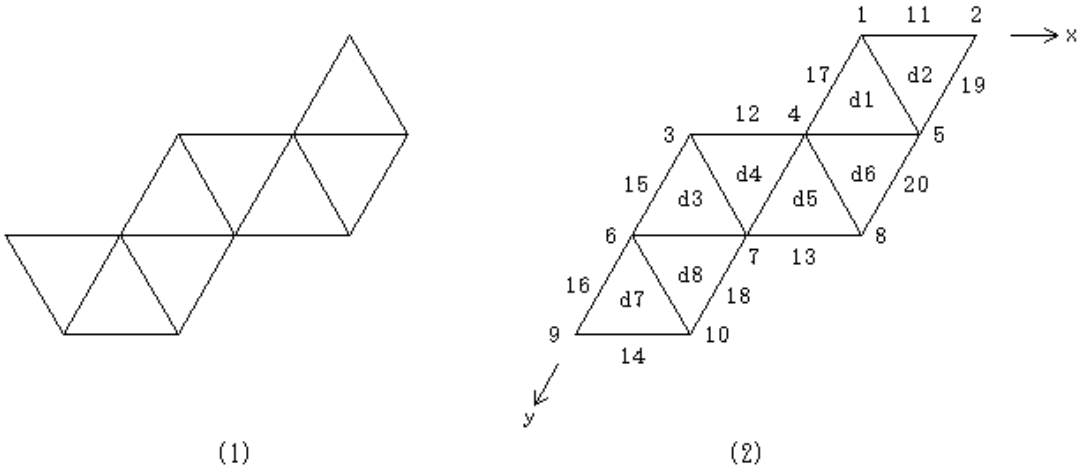
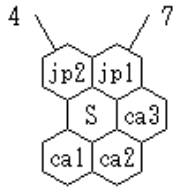


図 1

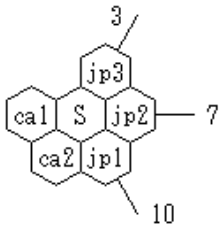
2. 近傍図

　図 2 は頂点、辺に関する近傍図です。正八面体では頂点の周りに正三角形が四つあります。したがって、図 1 の各頂点には 0〜3 個の正三角形が付加されます。前回述べた理由により、vertex 5、6 では実質的には付加が不要です。



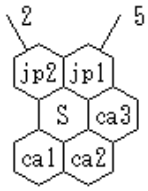
S:1(3)
jp1:(1,n)
jp2:(n-1,1) <=> ca1

vertex 1



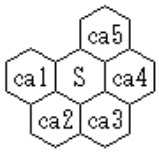
S:2(6)
jp1:(1,2n-1)
jp2:(1,2n-2)
jp3:(0,2n-3) <=> ca1

vertex 2



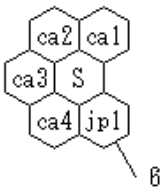
S:3(1)
jp1:(n,1)
jp2:(n,0) <=> ca1

vertex 3



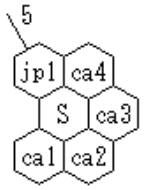
S:4

vertex 4



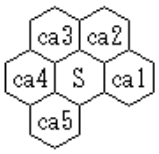
S:5(9)
jp1:(0,3n-4) <=> ca1

vertex 5



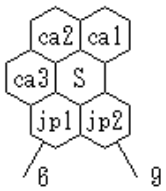
S:6(2)
jp1:(2(n-1),1) <=> ca1

vertex 6



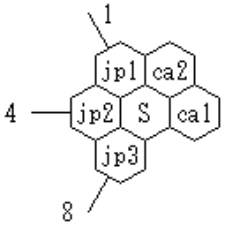
S:7

vertex 7



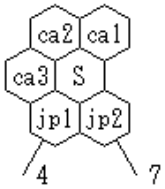
S:8(10)
jp1:(n-2,3n-4)
jp2:(n-2,3(n-1)) <=> ca1

vertex 8



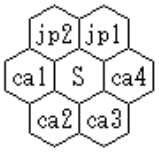
S:9(5)
jp1:(2n-3,n-2)
jp2:(2n-3,n-1)
jp3:(2(n-1),n) <=> ca1

vertex 9



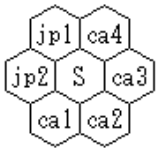
S:10(8)
jp1:(2n-3,2n-3)
jp2:(n-1,3n-4) <=> ca1

vertex 10



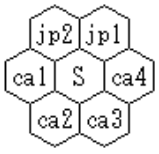
S:(x,0)
jp1:(1,x+1)
jp2:(1,x)

side 11



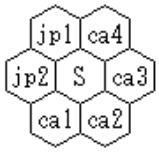
S:(0,y)
jp1:(y,1)
jp2:(y+1,1)

side 15



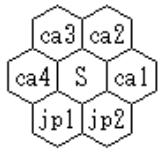
S:(x,n-1)
jp1:(n,x+1)
jp2:(n,x)

side 12

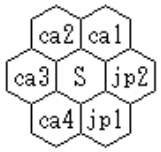


S:(n-1,y)
jp1:(y,n)
jp2:(y+1,n)

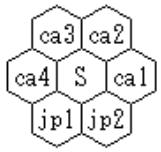
side 17



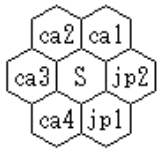
S:(x,2(n-1))
jp1:(n-2,x+n-2)
jp2:(n-2,x+(n-1))
side 13



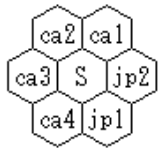
S:(n-1,y)
jp1:(y-(n-1),2n-3)
jp2:(y-n,2n-3)
side 18



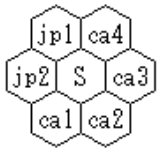
S:(x,3(n-1))
jp1:(2n-3,x+n-2)
jp2:(2n-3,x+(n-1))
side 14



S:(2(n-1),y)
jp1:(y-(n-1),3n-4)
jp2:(y-n,3n-4)
side 20



S:(2(n-1),y)
jp1:(1,y+1+2(n-1))
jp2:(1,y+2(n-1))
side 19



S:(0,y)
jp1:(2n-3,y-1-2(n-1))
jp2:(2n-3,y-2(n-1))
side 16

図 2

3. シードポイント

図 1-(2) に示される正八面体にシードポイントを配します。

(1) SP(3 × 1):3(d1)

d1 はシードポイントが配される正三角形の番号です。ある正三角形内に三つのシードポイントを配すると別のある正三角形内にも三つの対称なグラフィクスを得ます。この二つの正三角形を収束面と称します。

(2) SP(3 × 2):3(d1)+3(d8):d?-c

六つのシードポイントを二つの正三角形に等配する場合は二つの収束面を選択します。d?-c は互いに収束面であることを表します。

(3) SP(3 × 4):3(d1)+3(d3)+3(d5)+3(d7):d?-v, CP(all)

d?-v は正三角形が頂点で接することを表します。CP は閉じるべき頂点を表します。

(4) SP(3 × 8):3(d1)+3(d2)+,,,+3(d7)+3(d8):d?-s, CP(all)

d?-s は正三角形が辺で接することを表します。

(5) SP(1 × 2):1(d3)+1(d6):d?-c

(6) SP(1 × 2):1(d3)+1(d4):d?-s

(7) SP(1 × 2):1(d1)+1(d5):d?-v, CP(2):4, 6;2

(8) SP(1 × 4):1(d1)+1(d4)+1(d5)+1(d6):d?-sv, CP(2):4, 6;2

d?-sv は正三角形が辺で接していて共通の頂点の周りにあることを表します。

(9) SP(1 × 3):1(d2)+1(d4)+1(d6):d?-vr
d?-vr は正三角形が頂点で接していてリング状になっていることを表します。

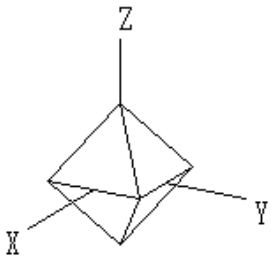
シードポイントと次の点 (塗番号 2) の座標は対称性を考慮して決められます。シードポイントと塗番号 2 が作るベクトルを第一ベクトルと称します。一塗点の第一ベクトルが与えられれば、他塗点のシードポイントと塗番号 2 は対称軸の周りのこの第一ベクトルの回転によって求められます。図 3 において Z 軸上に頂点 4, 6;2 があるとします。たとえば、(7) SP(1 × 2) の対称軸は Z 軸であり、(6) SP(1 × 2) と (5) SP(1 × 2) の対称軸は他の二軸です。

4. 収束面

収束面は複数の塗点が対称なグラフィクスを描き、最接近する正三角形です。この定義を拡張して一つの正三角形ではなく複数の塗点によって対称なグラフィクスが描かれる正三角形の集まりも収束面と称します。(7) SP(1 × 2) と (8) SP(1 × 4) のシードポイントは図 3 の上半分にあるとします。対称なグラフィクスが上半分と下半分に描かれるので、収束面は上半分と下半分です。したがって、両収束面にシードポイントを配した SP(1 × 2 × 2):1(d1)+1(d5)+1(d3)+1(d7):(d?-v) × 2, CP(2):4, 6;2 と SP(1 × 4 × 2):1(d1)+1(d4)+1(d5)+1(d6)+1(d2)+1(d3)+1(d7)+1(d8):(d?-sv) × 2, CP(all) においても対称なグラフィクスが両収束面に描かれます。

(9) SP(1 × 3):d?-vr においても収束面が存在します。これはかみ合いクラッチのような形をしています。両収束面にシードポイントを配した SP(1 × 3 × 2):1(d2)+1(d4)+1(d6)+1(d3)+1(d5)+1(d7):(d?-vr) × 2 においても対称なグラフィクスが両収束面に描かれます。

収束面はシードポイントを網羅するために考えられた正三角形の集まりです。一般的な幾何学的対称性に基づいて得られるシードポイントには限りがあります。実際に多面体上の対称なグラフィクスを探して様々な収束面を設けることによってシードポイントを搾り出すことができます。



#define _6dRow 1

図 3

5. 正四面体未満による領域の加工

正八面体の四つの正三角形を正四面体未満で置き換えると図 4 のような多面体が得られます。この多面体の作り方を図 5 に示します。最初は正三角形 R が四つあります。R を一つずつ置換図形で置き換えると最終的に正三角形は 16 になります。図中、線分の対応は色と矢印で表しています。連結していない二線分の色はユニークになっていて、二線分の色は適宜変えられます。(8) は全体的に反転・回転を施すと (7) と一致します。

頂点は以下のように整理されます。

・周りにある正三角形の数=3 である頂点：2, 4, 12, 14
・周りにある正三角形の数=6 である頂点：7, 8, 9, 6;10, 1;3;5, 11;13;15

辺の変換はすべて反転です。

・16⇔13, 20⇔21：$x = 2(n-1)$ に関する反転
・18⇔19, 22⇔23：$x = 6(n-1)$ に関する反転
・24⇔26, 25⇔27：$x = 4(n-1)$ に関する反転

シードポイントはたとえば SP(1 × 2):1(d1)+1(d12):d?-v, CP(2):7, 9 とします。図 4 の多面体は全体的に見ると正四面体ですから、正四面体においては SP(1 × 2):1(d1)+1(d2):d?-s が可能であることが推論できます。

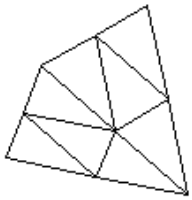


図 4

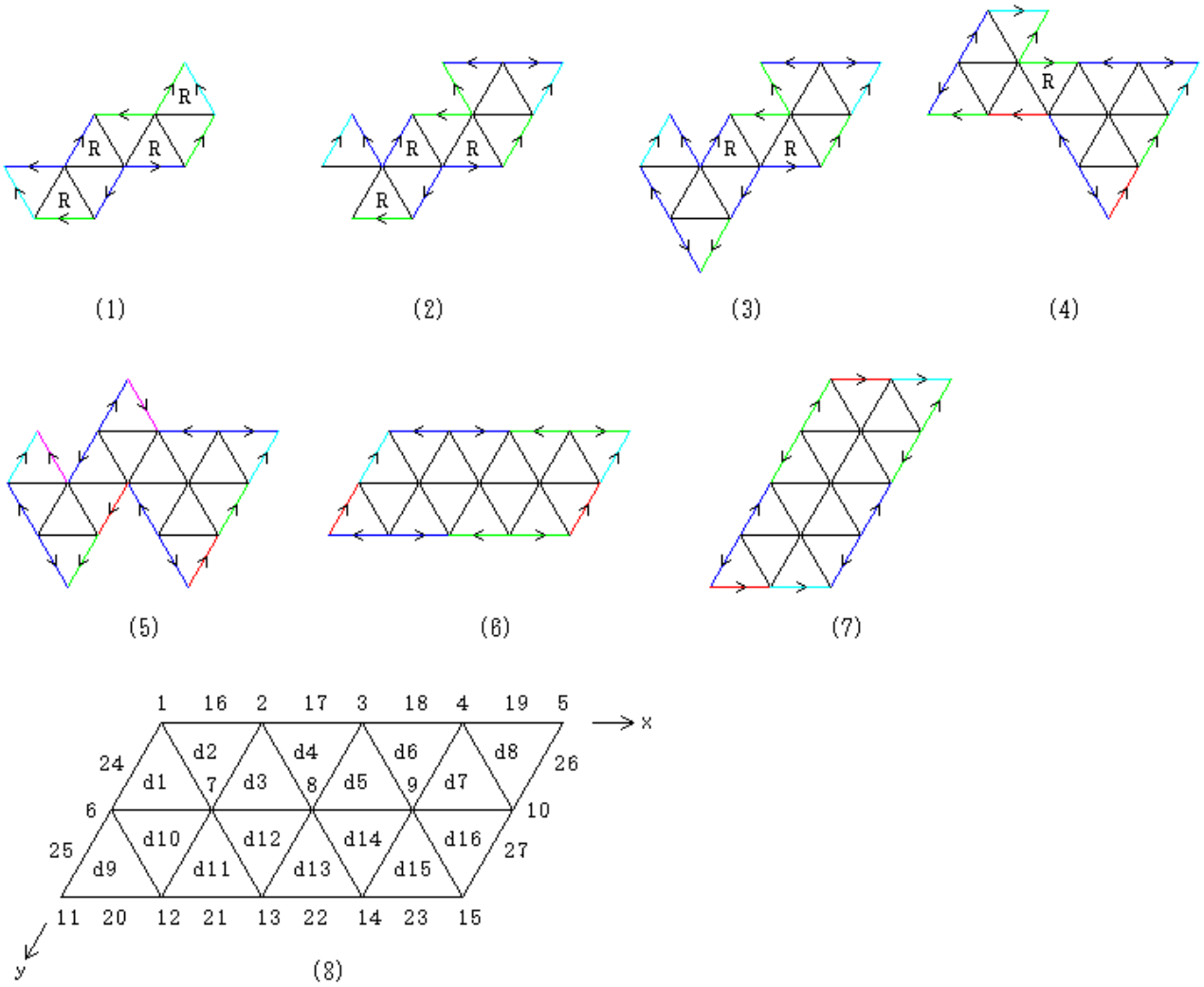


図 5

6. 正八面体未満による領域の加工

正八面体をいくつか連結すると図 6 のような多面体が得られます。図 6-(1) は正八面体の一つの正三角形を展開図が図 7 である図形 (正八面体未満) で置き換えることによって得られます。この多面体の作り方を図 8 に示します。正三角形 R を置換図形で置き換えると正三角形は 14 になります。

・周りにある正三角形の数=4 である頂点：2, 4, 11, 13, 1;3;5, 10;12;14
・周りにある正三角形の数=6 である頂点：7, 8, 6;9

辺の変換はすべて反転です。

・23⇔25, 24⇔26：$x = 3(n-1)/2$ に関する反転
・15⇔16：$y = x$ に関する反転
・21⇔22：$y = x + (n-1)$ に関する反転
・17⇔18：$x = 2(n-1)$ に関する直交座標的反転
・19⇔20：$x = n - 1$ に関する直交座標的反転

シードポイントはたとえば SP(3 × 1):3(d1) とします。この場合の最小収束面は d1、d14 であり、図 6-(1) をトンネルと考えると、これらはトンネルの二つの入り口です。

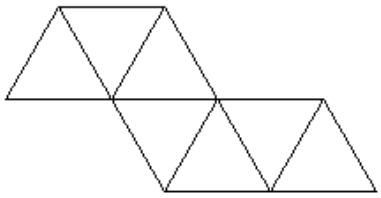
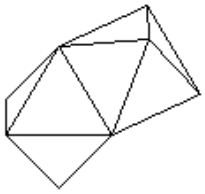
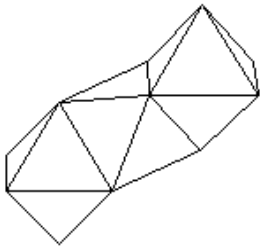


図 6



#define _6dRow 2

(1)



#define _6dRow 3

(2)

図 7

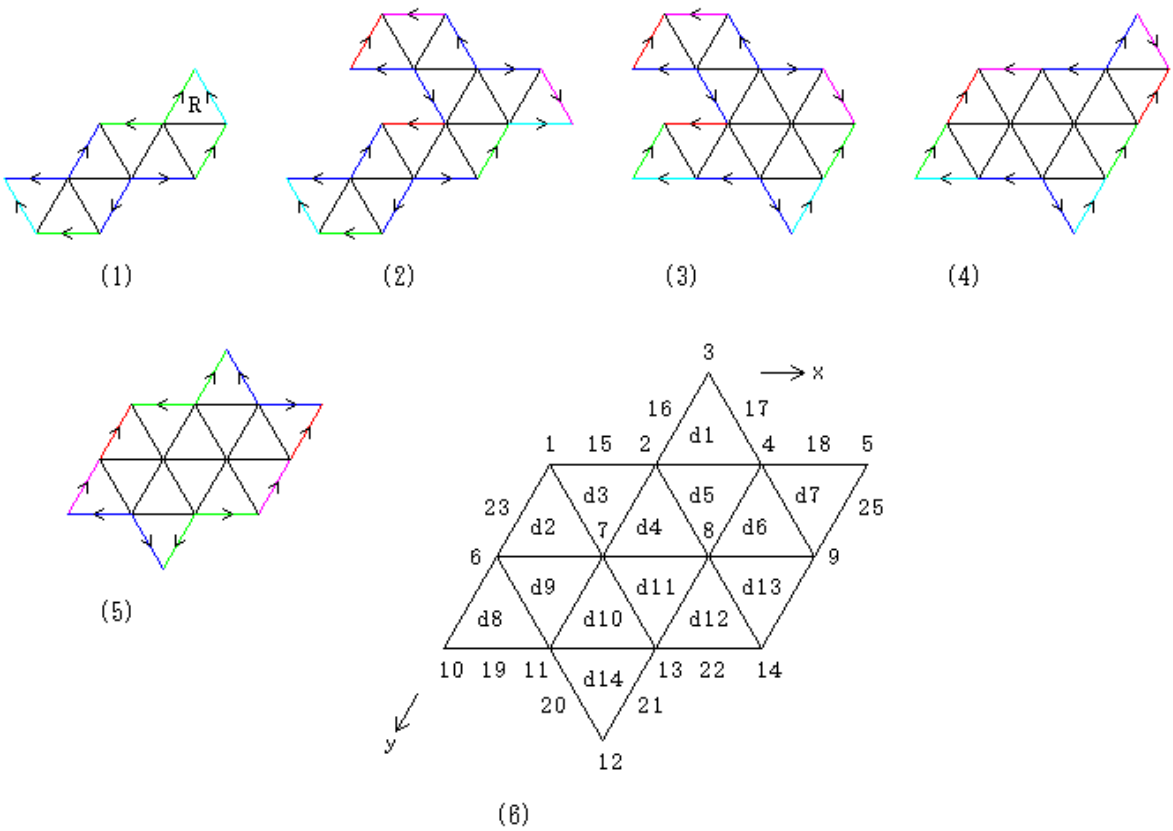


図 8

7. 課題

・図 4、図 7 の図形の頂点、辺に関する近傍図を作成してみてください。

・図 7 の図形のトンネルの方向の収束面を求めてみてください。正八面体、図 7-(2) の図形を奇トンネル、図 7-(1) の図形を偶トンネルと称します。

8. 具体例

前出のシードポイントの詳細なデータは以下の通りです。

(1) SP(3×1):3(d1)

・$n = 12$

・塗番号 1 の座標：塗点 a:$(n-1+2,\ 4)$、塗点 b:$(n-1+n-5,\ n-3)$、塗点 c:$(n-1+2,\ n-3)$

・塗番号 2 の座標：塗点 a:$\Delta y = 1$、塗点 b:$\Delta x = -1; \Delta y = -1$、塗点 c:$\Delta x = 1$

(2) SP(3×2):3(d1)+3(d8):d?-c

・$n = 12$

・塗番号 1 の座標：塗点 a_0:$(n-1+2,\ 4)$、塗点 b_0:$(n-1+n-5,\ n-3)$、塗点 c_0:$(n-1+2,\ n-3)$

・塗番号 1 の座標：塗点 a_1':$(4,\ 2(n-1)+2)$、塗点 b_1':$(n-3,\ 2(n-1)+2)$、塗点 c_1':$(n-3,\ 3(n-1)-4)$

・塗番号 2 の座標：塗点 a_0:$\Delta y = 1$、塗点 b_0:$\Delta x = -1; \Delta y = -1$、塗点 c_0:$\Delta x = 1$

・塗番号 2 の座標：塗点 a_1':$\Delta x = 1; \Delta y = 1$、塗点 b_1':$\Delta x = -1$、塗点 c_1':$\Delta y = -1$

(3) SP(3×4):3(d1)+3(d3)+3(d5)+3(d7):d?-v, CP(all)

・$n = 12$

・塗番号 1 の座標：塗点 a_0:$(n-1+2,\ 4)$、塗点 b_0:$(n-1+n-5,\ n-3)$、塗点 c_0:$(n-1+2,\ n-3)$；塗点 a_i、塗点 b_i、塗点 $c_i (i > 0)$:d1(a_0、b_0、c_0) の平行移動

・塗番号 2 の座標：塗点 a_i:$\Delta y = 1$、塗点 b_i:$\Delta x = -1$;$\Delta y = -1$、塗点 c_i:$\Delta x = 1$

(4) SP(3 × 8):3(d1)+3(d2)+,,,+3(d7)+3(d8):d?-s, CP(all)
・$n = 12$
・塗番号 1 の座標：塗点 a_0:$(n-1+2,\ 4)$、塗点 b_0:$(n-1+n-5,\ n-3)$、塗点 c_0:$(n-1+2,\ n-3)$；塗点 a_i、塗点 b_i、塗点 $\mathrm{c}_i(0 < i < 4)$:d1(a_0、b_0、c_0) の平行移動
・塗番号 1 の座標：塗点 a'_4:$(n-1+4,\ 2)$、塗点 b'_4:$(n-1+n-3,\ 2)$、塗点 c'_4:$(n-1+n-3,\ n-5)$；塗点 a'_i、塗点 b'_i、塗点 $\mathrm{c}'_i(i > 4)$:d2(a'_4、b'_4、c'_4) の平行移動
・塗番号 2 の座標：塗点 a_i:$\Delta y = 1$、塗点 b_i:$\Delta x = -1$;$\Delta y = -1$、塗点 c_i:$\Delta x = 1 \quad (i < 4)$
・塗番号 2 の座標：塗点 a'_i:$\Delta x = 1$;$\Delta y = 1$、塗点 b'_i:$\Delta x = -1$、塗点 c'_i:$\Delta y = -1 \quad (i > 3)$

(5) SP(1 × 2):1(d3)+1(d6):d?-c
・$n = 12$
・塗番号 1 の座標：塗点 c_0:$(2,\ n-1+n-3)$
・塗番号 1 の座標：塗点 a'_1:$(n-1+4,\ n-1+2)$
・塗番号 2 の座標：塗点 c_0:$\Delta y = -1$
・塗番号 2 の座標：塗点 a'_1:$\Delta x = 1$

(6) SP(1 × 2):1(d3)+1(d4):d?-s
・$n = 12$
・塗番号 1 の座標：塗点 c_0:$(2,\ n-1+n-3)$
・塗番号 1 の座標：塗点 b'_1:$(n-3,\ n-1+2)$
・塗番号 2 の座標：塗点 c_0:$\Delta y = -1$
・塗番号 2 の座標：塗点 b'_1:$\Delta y = 1$

(7) SP(1 × 2):1(d1)+1(d5):d?-v, CP(2):4, 6;2
・$n = 12$
・塗番号 1 の座標：塗点 c_0:$(n-1+2,\ n-3)$
・塗番号 1 の座標：塗点 a_1:$(n-1+2,\ n-1+4)$
・塗番号 2 の座標：塗点 c_0:$\Delta x = 1$
・塗番号 2 の座標：塗点 a_1:$\Delta y = 1$

(8) SP(1 × 4):1(d1)+1(d4)+1(d5)+1(d6):d?-sv, CP(2):4, 6;2
・$n = 12$
・塗番号 1 の座標：塗点 c_0:$(n-1+2,\ n-3)$
・塗番号 1 の座標：塗点 b'_1:$(n-3,\ n-1+2)$
・塗番号 1 の座標：塗点 a_2:$(n-1+2,\ n-1+4)$
・塗番号 1 の座標：塗点 a'_3:$(n-1+4,\ n-1+2)$
・塗番号 2 の座標：塗点 c_0:$\Delta x = 1$
・塗番号 2 の座標：塗点 b'_1:$\Delta x = -1$
・塗番号 2 の座標：塗点 a_2:$\Delta y = 1$
・塗番号 2 の座標：塗点 a'_3:$\Delta x = 1$;$\Delta y = 1$

(9) SP(1 × 3):1(d2)+1(d4)+1(d6):d?-vr
・$n = 12$
・塗番号 1 の座標：塗点 a'_0:$(n-1+4,\ 2)$

・塗番号 1 の座標：塗点 b'_1:$(n-1-2,\ n-1+2)$
・塗番号 1 の座標：塗点 b'_2:$(2(n-1)-2,\ n-1+2)$
・塗番号 2 の座標：塗点 a'_0:$\Delta x=1;\Delta y=1$
・塗番号 2 の座標：塗点 b'_1:$\Delta x=-1$
・塗番号 2 の座標：塗点 b'_2:$\Delta x=-1$

図 9 は図 2 による対称グラフィクスであり、以下はそのデータです。

・SP(3×1):3(d1)
・$n=12$
・塗番号 1 の座標：塗点 a:$(n-1+2,\ 4)$、塗点 b:$(n-1+n-5,\ n-3)$、塗点 c:$(n-1+2,\ n-3)$
・塗番号 2 の座標：塗点 a:$\Delta y=1$、塗点 b:$\Delta x=-1;\Delta y=-1$、塗点 c:$\Delta x=1$

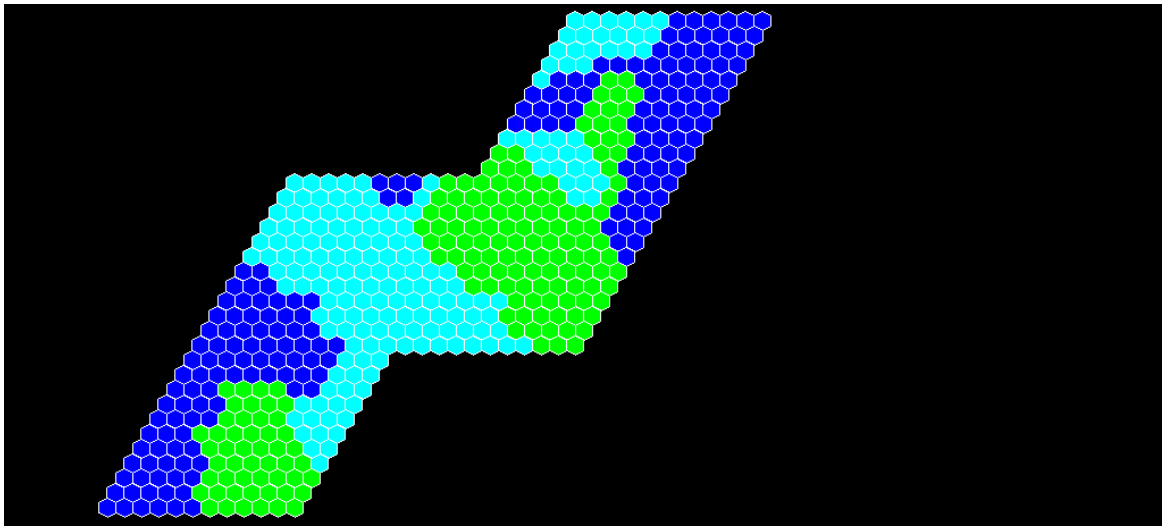


図 9

図 10 は図 5 による対称グラフィクスであり、以下はそのデータです。

・$n=12$
・SP(1×2):1(d1)+1(d12):d?-v, CP(2):7, 9
・塗番号 1 の座標：塗点 b_0:$(n-5,\ n-3)$
・塗番号 1 の座標：塗点 a'_1:$(n-1+4,\ n-1+2)$
・塗番号 2 の座標：塗点 b_0:$\Delta x=-1$
・塗番号 2 の座標：塗点 a'_1:$\Delta x=1$

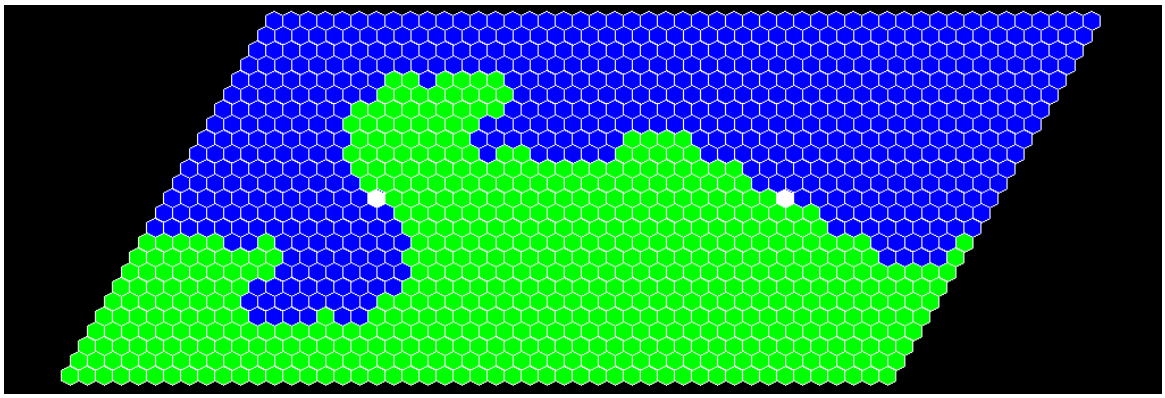


図 10

図 11 は図 8 による対称グラフィクスであり、以下はそのデータです。

・SP(3×1):3(d1)
・$n = 12$
・塗番号 1 の座標：塗点 a:$(n-1+2,\ 4)$、塗点 b:$(n-1+n-5,\ n-3)$、塗点 c:$(n-1+2,\ n-3)$
・塗番号 2 の座標：塗点 a:$\Delta y = 1$、塗点 b:$\Delta x = -1; \Delta y = -1$、塗点 c:$\Delta x = 1$

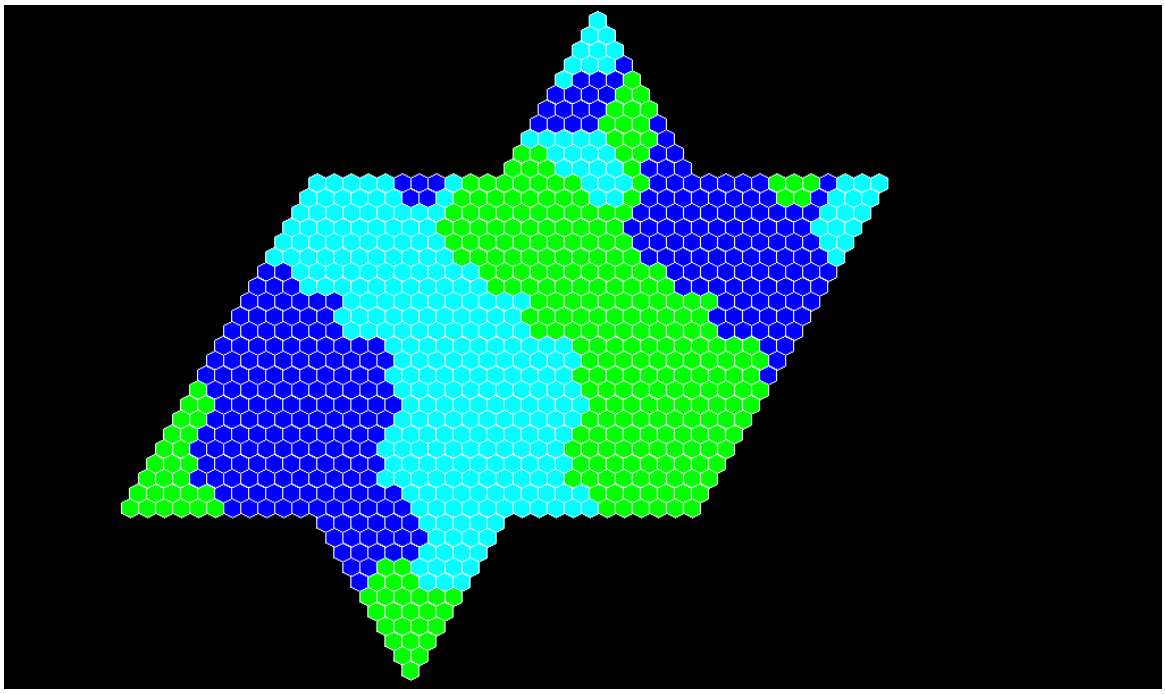


図 11

****

List 1:cag_3.c

```
/* t2.22                                                                    */
/* 2018 Morio Kikuchi                                                       */

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

#define VGACOLORS /*16*/50
#define GKS GetKeyState
#define ASIZE_MS ((1024*768)/CPMAX)
#define CPMAX /*3*/6
#define X0 (330+20)
#define Y0 (10)
#define dyMAX 603
#define RESO 12
#define _6dRow 2

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

char refill,pauseflag,fieldflag,jmpflag;
char charcode,charflag;
int Ca,C1,C2,C3,C4,C5,C7,x[8],y[8],x_[8],y_[8],cc_sum[VGACOLORS];
int X,Y,X_,Y_,Nx,Ny,Nxp,Nxm,Nyp,Nym;
int algo,combination,drn,ig,PIXSIZE;
int xt,xtm,xtp,yt,_6d,ssize,std_x,std_y,last_x,last_y;
long asize=ASIZE_MS;
long rcount[CPMAX];

/*unsigned */char **pixel;
int DS[][2]={{640,480},{640,480},{800,600},{1024,768},{1280,1024},{1600,1200}};
long fp_mem[CPMAX];
double hexagon_x[6],hexagon_y[6];

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_;} 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),field(void),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);
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)
{
long mytime;

WB=1;
refill=1;

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

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

xt=3*(RESO-1);
_6d=(1+_6dRow)*(RESO-1);
yt=_6d+(RESO-1);
```

```
if(setup_()==1) return 1;

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

combination=1;
drn=4;

arrayreset();
field();
cag_r();

while(1){
check_rcount();
printf(" \n");

if(refill==0) break;
beep(50);

delay_(6000);
if(pauseflag==1) {pauseflag=0;use_subroop();}
if(refill==0) break;

/*drn=random_(6);*/

field();
cag_r();
}/**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);
}

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);
}
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+1.7)*(sqrt(3)/2)*PIXSIZE;
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 setup_(void)
{
int i,dy;

if(_6dRow<1) return 1;

PIXSIZE=15;
dy=Y0+(yt+1.7)*(sqrt(3)/2)*PIXSIZE;

if(dy>dyMAX){
while(1){
PIXSIZE--;
dy=Y0+(yt+1.7)*(sqrt(3)/2)*PIXSIZE;
if(dy<=603) break;
}
}

if(PIXSIZE<4) PIXSIZE=4;

hexagon_x[0]=0.;
hexagon_x[1]=ff_fc(0.5*PIXSIZE);
```

```
hexagon_x[2]=ff_fc(1.*PIXSIZE);
hexagon_x[3]=ff_fc(1.*PIXSIZE);
hexagon_x[4]=ff_fc(0.5*PIXSIZE);
hexagon_x[5]=ff_fc(0.*PIXSIZE);

hexagon_y[0]=0.;
hexagon_y[1]=ff_fc((-sqrt(3)/6)*PIXSIZE);
hexagon_y[2]=0.;
hexagon_y[3]=ff_fc((sqrt(3)/3)*PIXSIZE);
hexagon_y[4]=ff_fc((sqrt(3)/2)*PIXSIZE);
hexagon_y[5]=ff_fc((sqrt(3)/3)*PIXSIZE);

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

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

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==XRESO) break;
}

return 0;
}/** setup_ **/


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

```
setup();

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+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=BeginPaint(hwnd,&paintstruct);*/
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==XRESO) 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=7;i<9;i++){                    /* 7, 8 */
irgb[i].red=128+32*(9-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;
}
}/** initpalette **/


void BitBlt_full(void)
{
bitblt(1,0,0,XRESO,YRESO,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==YRESO) break;
}

i++;
if(i==XRESO) break;
}
}/** arrayreset **/


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

if(nx<0 || nx>3*(RESO-1)) return 0;
if(nx<1*(RESO-1) && ny<1*(RESO-1)) return 0;
if(ny<nx-(RESO-1) && ny<1*(RESO-1)) return 0;
if(ny>nx+_6dRow*(RESO-1) && ny>_6d) return 0;
if(nx>2*(RESO-1) && ny>_6d) return 0;

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

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

i=0;
while(1){
vertex[i].x=hexagon_x[i]+dx;
vertex[i].y=hexagon_y[i]+dy;
i++;if(i==6) break;
}

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);

pixel[nx][ny]=pcolor;

return 0;
}/** putpixel **/



void check_rcount(void)
{
int i,j,k,m,n,dx,dy,Li,Lj;

Li=3;Lj=4;
```

```
for(j=0;j<Lj;j++)
for(i=0;i<Li;i++){
dx=(RESO-1)*i;
dy=(RESO-1)*j;

/* left */
if(pixel[dx][dy]!=0 && pixel[dx+RESO-1][dy+RESO-1]!=0 && pixel[dx][dy+RESO-1]!=0){
for(k=0;k<VGACOLORS;k++)
cc_sum[k]=0;

for(m=0;m<RESO;m++)
for(n=0;n<RESO;n++) {if(m>=n && pixel[dx+n][dy+m]>0) cc_sum[pixel[dx+n][dy+m]]++;}

if(/*(cc_sum[9]==cc_sum[10])&&(cc_sum[9]==cc_sum[11])*/1){
printf(" i=%d j=%d left\n",i,j);

for(k=0;k<VGACOLORS;k++)
if(cc_sum[k]) printf(" cc%2d:%ld\n",k,cc_sum[k]);
}
}

/* right */
if(pixel[dx][dy]!=0 && pixel[dx+RESO-1][dy+RESO-1]!=0 && pixel[dx+RESO-1][dy]!=0){
for(k=0;k<VGACOLORS;k++)
cc_sum[k]=0;

for(m=0;m<RESO;m++)
for(n=0;n<RESO;n++) {if(m<=n && pixel[dx+n][dy+m]>0) cc_sum[pixel[dx+n][dy+m]]++;}

if(/*(cc_sum[9]==cc_sum[10])&&(cc_sum[9]==cc_sum[11])*/1){
printf(" i=%d j=%d right\n",i,j);

for(k=0;k<VGACOLORS;k++)
if(cc_sum[k]) printf(" cc%2d:%ld\n",k,cc_sum[k]);
}
}
}/**for(i)**/

for(i=0;i<CPMAX;i++)
printf(" %ld\n",rcount[i]);
}/** check_rcount **/


void field_rect(int x,int y,int dx,int dy)
{
```

```
int i,j;

fieldflag=1;

for(j=y;j<y+dy;j++)
for(i=x;i<x+dx;i++){
putpixel_(i,j,15);
}

fieldflag=0;
}/** field_rect **/


void field_trian(int lr,int x,int y)
{
int m,n;

fieldflag=1;

for(m=0;m<RESO;m++)
for(n=0;n<RESO;n++){
if(lr==0) {if(m>=n) putpixel_(x+n,y+m,15);}
else      {if(m<=n) putpixel_(x+n,y+m,15);}
}

fieldflag=0;
}/** field_trian **/


void field(void)
{
field_rect(0,RESO-1,3*(RESO-1)+1,_6dRow*(RESO-1)+1);
field_trian(0,RESO-1,0);
field_trian(1,RESO-1,_6d);
}/** field **/


void putpixel_(int nx,int ny,int pcolor)
{
char flag;
int tmp,dlt;

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

```
if((nx==RESO-1 || nx==2*(RESO-1)) && ny==RESO-1){
/* 2, 4 */
}
else if((nx==RESO-1 || nx==2*(RESO-1)) && ny==_6d){
/* 11, 13 */
}
else if((nx==RESO-1 && ny==0)||((nx==0 || nx==xt) && ny==RESO-1)){
/* 1;3;5 */
putpixel(RESO-1,0,pcolor);
putpixel(0,RESO-1,pcolor);
putpixel(xt,RESO-1,pcolor);
}
else if((nx==2*(RESO-1) && ny==yt)||((nx==0 || nx==xt) && ny==_6d)){
/* 10;12;14 */
putpixel(2*(RESO-1),yt,pcolor);
putpixel(0,_6d,pcolor);
putpixel(xt,_6d,pcolor);
}
else if(/*(nx==0 || nx==xt) && ny==2*(RESO-1)*/0){
/* 6;9 */
putpixel(0,2*(RESO-1),pcolor);
putpixel(xt,2*(RESO-1),pcolor);
}
else{
if((nx==0 || nx==xt) && (ny>=RESO && ny<=_6d-1))   flag=1;  /* 23, 25, 24, 26 */
else if(nx==RESO-1 && (ny>=1 && ny<=RESO-2))          flag=2;  /* 16 */
else if((nx>=1 && nx<=RESO-2) && ny==RESO-1)          flag=2;  /* 15 */
else if((nx>=2*RESO-1 && nx<=xt-1) && ny==_6d) flag=3;  /* 22 */
else if(nx==2*(RESO-1) && (ny>=_6d+1 && ny<=yt-1)) flag=3;  /* 21 */
else if((nx>=RESO && nx<=2*RESO-3) && ny==nx-(RESO-1))   flag=4;  /* 17 */
else if((nx>=2*RESO-1 && nx<=xt-1) && ny==RESO-1)        flag=5;  /* 18 */
else if((nx>=RESO && nx<=2*RESO-3) && ny==nx+_6dRow*(RESO-1)) flag=6;  /* 20 */
else if((nx>=1 && nx<=RESO-2) && ny==_6d)         flag=7;  /* 19 */
else flag=-1;

if(flag==0){
}
else if(flag>0){
if(flag==1){
nx=xt-nx;
}
else if(flag==2){
tmp=nx;nx=ny;ny=tmp;
}
else if(flag==3){
```

```
tmp=nx;nx=ny;ny=tmp;
nx-=(_6dRow-1)*(RESO-1);ny+=(_6dRow-1)*(RESO-1);
}
else if(flag==4){
nx=4*(RESO-1)-nx;ny=RESO-1;
}
else if(flag==5){
dlt=3*(RESO-1)-nx;nx=RESO-1+dlt;ny=dlt;
}
else if(flag==6){
nx=2*(RESO-1)-nx;ny=_6d;
}
else if(flag==7){
dlt=RESO-1-nx;nx=RESO-1+dlt;ny=_6d+dlt;
}

putpixel(nx,ny,pcolor);
}
}
}/** putpixel_ **/


void putpixel_2(int nx,int ny,int pcolor)
{
char flag;
int tmp,dlt;

X_=nx;Y_=ny;

if((nx==RESO-1 || nx==2*(RESO-1)) && ny==RESO-1){
/* 2, 4 */
}
else if((nx==RESO-1 || nx==2*(RESO-1)) && ny==_6d){
/* 11, 13 */
}
else if((nx==RESO-1 && ny==0)||((nx==0 || nx==xt) && ny==RESO-1)){
/* 1;3;5 */
}
else if((nx==2*(RESO-1) && ny==yt)||((nx==0 || nx==xt) && ny==_6d)){
/* 10;12;14 */
}
else if(/*(nx==0 || nx==xt) && ny==2*(RESO-1)*/0){
/* 6;9 */
}
else{
if((nx==0 || nx==xt) && (ny>=RESO && ny<=_6d-1))   flag=1;  /* 23, 25, 24, 26 */
```

```
else if(nx==RESO-1 && (ny>=1 && ny<=RESO-2))          flag=2;  /* 16 */
else if((nx>=1 && nx<=RESO-2) && ny==RESO-1)          flag=2;  /* 15 */
else if((nx>=2*RESO-1 && nx<=xt-1) && ny==_6d) flag=3;  /* 22 */
else if(nx==2*(RESO-1) && (ny>=_6d+1 && ny<=yt-1)) flag=3;  /* 21 */
else if((nx>=RESO && nx<=2*RESO-3) && ny==nx-(RESO-1))   flag=4;  /* 17 */
else if((nx>=2*RESO-1 && nx<=xt-1) && ny==RESO-1)        flag=5;  /* 18 */
else if((nx>=RESO && nx<=2*RESO-3) && ny==nx+_6dRow*(RESO-1)) flag=6;  /* 20 */
else if((nx>=1 && nx<=RESO-2) && ny==_6d)         flag=7;  /* 19 */
else flag=-1;

if(flag==0){
}
else if(flag>0){
if(flag==1){
nx=xt-nx;
}
else if(flag==2){
tmp=nx;nx=ny;ny=tmp;
}
else if(flag==3){
tmp=nx;nx=ny;ny=tmp;
nx-=(_6dRow-1)*(RESO-1);ny+=(_6dRow-1)*(RESO-1);
}
else if(flag==4){
nx=4*(RESO-1)-nx;ny=RESO-1;
}
else if(flag==5){
dlt=3*(RESO-1)-nx;nx=RESO-1+dlt;ny=dlt;
}
else if(flag==6){
nx=2*(RESO-1)-nx;ny=_6d;
}
else if(flag==7){
dlt=RESO-1-nx;nx=RESO-1+dlt;ny=_6d+dlt;
}

X_=nx;Y_=ny;
}
}
}/** putpixel_2 **/


/* Figure 7 */
int getpixel_(int x,int y,int nx,int ny)
{
int flag;
```

```
int x_,y_,dlt;

/*if(nx<0 || nx>3*(RESO-1)) return 0;
if(nx<1*(RESO-1) && ny<1*(RESO-1)) return 0;
if(ny<nx-(RESO-1) && ny<1*(RESO-1)) return 0;
if(ny>nx+_6dRow*(RESO-1) && ny>_6d) return 0;
if(nx>2*(RESO-1) && ny>_6d) return 0;*/

if(x==RESO-1 && y==0)                    flag=3;
else if(x==0 && y==RESO-1)               flag=1;
else if(x==RESO-1 && y==RESO-1)          flag=2;
else if(x==2*(RESO-1) && y==RESO-1)      flag=4;
else if(x==xt && y==RESO-1)              flag=5;
/*else if(x==0 && y==2*(RESO-1))           flag=6;
else if(x==xt && y==2*(RESO-1))          flag=9;*/
else if(x==0 && y==_6d)           flag=10;
else if(x==RESO-1 && y==_6d)      flag=11;
else if(x==2*(RESO-1) && y==_6d) flag=13;
else if(x==xt && y==_6d)          flag=14;
else if(x==2*(RESO-1) && y==yt)          flag=12;

else if((x>=1 && x<=RESO-2) && y==RESO-1)            flag=15;
else if(x==RESO-1 && (y>=1 && y<=RESO-2))            flag=16;
else if((x>=RESO && x<=2*RESO-3) && y==x-(RESO-1))   flag=17;
else if((x>=2*RESO-1 && x<=xt-1) && y==RESO-1)       flag=18;
else if(x==0 && (y>=RESO && y<=_6d-1))            flag=23;
else if(x==xt && (y>=RESO && y<=_6d-1))           flag=25;
else if((x>=1 && x<=RESO-2) && y==_6d)        flag=19;
else if((x>=2*RESO-1 && x<=xt-1) && y==_6d)   flag=22;
else if((x>=RESO && x<=2*RESO-3) && y==x+_6dRow*(RESO-1)) flag=20;
else if(x==2*(RESO-1) && (y>=_6d+1 && y<=yt-1))   flag=21;
else flag=0;

X=nx;Y=ny;
jmpflag=0;
/*goto end;*/

if(flag==2){  /* 2 */
if(nx<1*(RESO-1) && ny<1*(RESO-1)) return 0;
}
else if(flag==4){  /* 4 */
if(ny<nx-(RESO-1) && ny<1*(RESO-1)) return 0;
}
else if(flag==11){  /* 11 */
if(ny>nx+_6dRow*(RESO-1) && ny>_6d) return 0;
}
```

```
else if(flag==13){  /* 13 */
if(nx>2*(RESO-1) && ny>_6d) return 0;
}
else if(flag==1){  /* 1 */

     if(nx==x && ny==y-1) {X=RESO;Y=1;jmpflag=2;}
else if(nx<0 || ny<RESO-1) return 0;
}
else if(flag==3){  /* 3 */

     if(nx==x+1 && ny==y) {X=xt;Y=RESO;jmpflag=4;}
else if(nx==x && ny==y-1) {X=1;Y=RESO;jmpflag=2;}
else if(nx<RESO-1 || ny<nx-(RESO-1)) return 0;
}
else if(flag==5){  /* 5 */

     if(nx==x+1 && ny==y+1) {X=1;Y=RESO;jmpflag=1;}
else if(nx==x+1 && ny==y) {X=1;Y=RESO-1;jmpflag=1;}
else if(nx>xt || ny<RESO-1) return 0;
}
else if(flag==10){  /* 10 */

     if(nx==x-1 && ny==y-1) {X=xt-1;Y=_6d-1;jmpflag=1;}
else if(nx==x-1 && ny==y) {X=xt-1;Y=_6d;jmpflag=1;}
else if(nx<0 || ny>_6d) return 0;
}
else if(flag==12){  /* 12 */

     if(nx==x-1 && ny==y) {X=0;Y=_6d-1;jmpflag=6;}
else if(nx==x && ny==y+1) {X=xt-1;Y=_6d-1;jmpflag=3;}
else if(nx>2*(RESO-1) || ny>nx+_6dRow*(RESO-1)) return 0;
}
else if(flag==14){  /* 14 */

     if(nx==x && ny==y+1) {X=2*RESO-3;Y=yt-1;jmpflag=3;}
else if(nx>xt || ny>_6d) return 0;
}
else if(flag==6 || flag==23 || flag==24){  /* 6, 23, 24 */

if(nx==-1) {X=xt-1;Y=ny;jmpflag=1;}
}
else if(flag==9 || flag==25 || flag==26){  /* 9, 25, 26 */

if(nx==xt+1) {X=1;Y=ny;jmpflag=1;}
}
else if(flag==15){  /* 15 */
```

```
     if(nx==x-1 && ny==RESO-2) {X=RESO;Y=x;jmpflag=2;}
else if(nx==x && ny==RESO-2)   {X=RESO;Y=x+1;jmpflag=2;}
}
else if(flag==16){  /* 16 */

     if(nx==RESO-2 && ny==y-1) {X=y;Y=RESO;jmpflag=2;}
else if(nx==RESO-2 && ny==y)   {X=y+1;Y=RESO;jmpflag=2;}
}
else if(flag==22){  /* 22 */

     if(nx==x && ny==y+1) {X=2*RESO-3;Y=x+(_6dRow-1)*(RESO-1)-1;jmpflag=3;}
else if(nx==x+1 && ny==y+1)   {X=2*RESO-3;Y=x+(_6dRow-1)*(RESO-1);jmpflag=3;}
}
else if(flag==21){  /* 21 */

     if(nx==2*RESO-1 && ny==y+1) {X=y-(_6dRow-1)*(RESO-1);Y=_6d-1;jmpflag=3;}
else if(nx==2*RESO-1 && ny==y)   {X=y-(_6dRow-1)*(RESO-1)-1;Y=_6d-1;jmpflag=3;}
}
else if(flag==17){  /* 17 */

x_=4*(RESO-1)-x;y_=RESO-1;
     if(nx==x+1 && ny==y) {X=x_;Y=y_+1;jmpflag=4;}
else if(nx==x && ny==y-1) {X=x_+1;Y=y_+1;jmpflag=4;}
}
else if(flag==18){  /* 18 */

dlt=3*(RESO-1)-x;x_=RESO-1+dlt;y_=dlt;
     if(nx==x && ny==y-1) {X=x_-1;Y=y_;jmpflag=5;}
else if(nx==x-1 && ny==y-1) {X=x_;Y=y_+1;jmpflag=5;}
}
else if(flag==20){  /* 20 */

x_=2*(RESO-1)-x;y_=_6d;
     if(nx==x && ny==y+1) {X=x_-1;Y=y_-1;jmpflag=6;}
else if(nx==x-1 && ny==y) {X=x_;Y=y_-1;jmpflag=6;}
}
#if 1
else if(flag==19){  /* 19 */

dlt=RESO-1-x;x_=RESO-1+dlt;y_=_6d+dlt;
     if(nx==x && ny==y+1) {X=x_+1;Y=y_;jmpflag=7;}
else if(nx==x+1 && ny==y+1) {X=x_;Y=y_-1;jmpflag=7;}
}
#endif
```

```
end:
return pixel[X][Y];
}/** getpixel_ **/
#if 0
/* Figure 4 */
int getpixel_(int x,int y,int nx,int ny)
{
int flag;

/*if(nx<0 || ny<0 || nx>4*(RESO-1) || ny>2*(RESO-1)) return 0;*/

if(x==0 && y==0)                    flag=1;
else if(x==RESO-1 && y==0)          flag=2;
else if(x==xtm && y==0)             flag=3;
else if(x==3*(RESO-1) && y==0)      flag=4;
else if(x==xt && y==0)              flag=5;
/*else if(x==0 && y==RESO-1)          flag=6;
else if(x==xt && y==RESO-1)         flag=10;*/
else if(x==0 && y==yt)              flag=11;
else if(x==RESO-1 && y==yt)         flag=12;
else if(x==xtm && y==yt)            flag=13;
else if(x==3*(RESO-1) && y==yt)     flag=14;
else if(x==xt && y==yt)             flag=15;

else if((x>=1 && x<=xtm-1) && y==0)     flag=16;
else if((x>=xtm+1 && x<=xt-1) && y==0)  flag=18;
else if((x>=1 && x<=xtm-1) && y==yt)    flag=20;
else if((x>=xtm+1 && x<=xt-1) && y==yt) flag=22;
else if(x==0 && (y>=1 && y<=yt-1))      flag=24;
else if(x==xt && (y>=1 && y<=yt-1))     flag=26;
else flag=0;

X=nx;Y=ny;
jmpflag=0;
/*goto end;*/

if(flag==1){  /* 1 */

     if(nx==x && ny==y-1) {X=xtm;Y=1;jmpflag=1;}
else if(nx==x-1 && ny==y-1) {X=xtm+1;Y=1;jmpflag=1;}
else if(nx==x-1 && ny==y) {X=xtm+1;Y=0;jmpflag=1;}
}
else if(flag==3){  /* 3 */

     if(nx==x && ny==y-1) {X=xt;Y=1;jmpflag=2;}
else if(nx==x-1 && ny==y-1) {X=1;Y=1;jmpflag=1;}
```

```
}
else if(flag==5){  /* 5 */

     if(nx==x+1 && ny==y+1) {X=1;Y=1;jmpflag=3;}
else if(nx==x+1 && ny==y) {X=1;Y=0;jmpflag=3;}
else if(nx==x && ny==y-1) {X=xtm;Y=1;jmpflag=2;}
else if(nx==x-1 && ny==y-1) {X=xtm+1;Y=1;jmpflag=2;}
}
else if(flag==2 || flag==4){  /* 2, 4 */
if(ny<0) return 0;
}
else if(flag==6){  /* 6 */

     if(nx==x-1 && ny==y-1) {X=xt-1;Y=/*RESO-2*//*y-1*/ny;jmpflag=3;}
else if(nx==x-1 && ny==y) {X=xt-1;Y=/*RESO-1*//*y*/ny;jmpflag=3;}
}
else if(flag==10){  /* 10 */

     if(nx==x+1 && ny==y+1) {X=1;Y=/*RESO*//*y+1*/ny;jmpflag=3;}
else if(nx==x+1 && ny==y) {X=1;Y=/*RESO-1*//*y*/ny;jmpflag=3;}
}
else if(flag==11){  /* 11 */

     if(nx==x-1 && ny==y-1) {X=xt-1;Y=yt-1;jmpflag=3;}
else if(nx==x-1 && ny==y) {X=xt-1;Y=yt;jmpflag=3;}
else if(nx==x && ny==y+1) {X=xtm;Y=yt-1;jmpflag=1;}
else if(nx==x+1 && ny==y+1) {X=xtm-1;Y=yt-1;jmpflag=1;}
}
else if(flag==13){  /* 13 */

     if(nx==x && ny==y+1) {X=0;Y=yt-1;jmpflag=1;}
else if(nx==x+1 && ny==y+1) {X=xt-1;Y=yt-1;jmpflag=2;}
}
else if(flag==15){  /* 15 */

     if(nx==x && ny==y+1) {X=xtm;Y=yt-1;jmpflag=2;}
else if(nx==x+1 && ny==y+1) {X=xtm-1;Y=yt-1;jmpflag=2;}
else if(nx==x+1 && ny==y) {X=1;Y=yt;jmpflag=3;}
}
else if(flag==12 || flag==14){  /* 12, 14 */
if(ny>yt) return 0;
}
else if(flag==16){  /* 16, 17 */
if(ny==-1) {X=xtm-nx;Y=1;jmpflag=1;}
}
else if(flag==18){  /* 18, 19 */
```

```
if(ny==-1) {X=xtp-nx;Y=1;jmpflag=2;}
}
else if(flag==20){  /* 20, 21 */
if(ny==yt+1) {X=xtm-nx;Y=yt-1;jmpflag=1;}
}
else if(flag==22){  /* 22, 23 */
if(ny==yt+1) {X=xtp-nx;Y=yt-1;jmpflag=2;}
}
else if(flag==24){  /* 24, 25, 6 */
if(nx==-1) {X=xt-1;Y=ny;jmpflag=3;}
}
else if(flag==26){  /* 26, 27, 10 */
if(nx==xt+1) {X=1;Y=ny;jmpflag=3;}
}

end:
return pixel[X][Y];
}/** getpixel_ **/
#endif


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 **/


double getangle(int next_x,int next_y)
{
char sign;
long product,p1,p2;
double val;

if(next_x==last_x && next_y==last_y) {beep(10000);refill=0;return 0;}

product=(next_x-std_x)*(std_y-last_y)-(next_y-std_y)*(std_x-last_x);
if(product==0) sign=0;
else if(product>0) sign=1;
else sign=-1;

if(sign==0) return 0;
else{
product=(next_x-std_x)*(std_x-last_x)+(next_y-std_y)*(std_y-last_y);
p1=(next_x-std_x)*(next_x-std_x)+(next_y-std_y)*(next_y-std_y);
p2=(std_x-last_x)*(std_x-last_x)+(std_y-last_y)*(std_y-last_y);
val=(double)product/(sqrt((double)p1)*sqrt((double)p2));

if(val>1) val=1;
else if(val<-1) val=-1;

return sign*acos(val);
}
}/** getangle **/


void mod_XY(int flag)
{
if(flag!=1) {putpixel_2(Nxp,Ny,0);x_[1]=X_;y_[1]=Y_;}
if(flag!=5) {putpixel_2(Nxp,Nyp,0);x_[5]=X_;y_[5]=Y_;}
if(flag!=2) {putpixel_2(Nx,Nyp,0);x_[2]=X_;y_[2]=Y_;}
if(flag!=3) {putpixel_2(Nxm,Ny,0);x_[3]=X_;y_[3]=Y_;}
if(flag!=7) {putpixel_2(Nxm,Nym,0);x_[7]=X_;y_[7]=Y_;}
if(flag!=4) {putpixel_2(Nx,Nym,0);x_[4]=X_;y_[4]=Y_;}
}/** mod_XY **/


int nv(int flag,int cc,int plc)
{
```

```
int order;

if(0) {if(cc==Ca) return 0;else return 1;}

order=1;

while(1){
if(flag==0){  /* for CW */
if(plc==1){
if(order==1){
mod_XY(5);
  if(C1==Ca) {if(x_[1]!=x[5] || y_[1]!=y[5]) return 0;}else if(C1!=0) return 1;}
else if(order==2){
  if(C4==Ca) {if(x_[4]!=x[5] || y_[4]!=y[5]) return 0;}else if(C4!=0) return 1;}
else if(order==3){
  if(C7==Ca) {if(x_[7]!=x[5] || y_[7]!=y[5]) return 0;}else if(C7!=0) return 1;}
else if(order==4){
  if(C3==Ca) {if(x_[3]!=x[5] || y_[3]!=y[5]) return 0;}else if(C3!=0) return 1;}
else if(order==5){
  if(C2==Ca) {if(x_[2]!=x[5] || y_[2]!=y[5]) return 0;}else if(C2!=0) return 1;}
else if(order==6){
  if(C5==Ca) {if(x_[5]!=x[5] || y_[5]!=y[5]) return 0;}else if(C5!=0) return 1;}
}
else if(plc==4){
if(order==1){
mod_XY(1);
  if(C4==Ca) {if(x_[4]!=x[1] || y_[4]!=y[1]) return 0;}else if(C4!=0) return 1;}
else if(order==2){
  if(C7==Ca) {if(x_[7]!=x[1] || y_[7]!=y[1]) return 0;}else if(C7!=0) return 1;}
else if(order==3){
  if(C3==Ca) {if(x_[3]!=x[1] || y_[3]!=y[1]) return 0;}else if(C3!=0) return 1;}
else if(order==4){
  if(C2==Ca) {if(x_[2]!=x[1] || y_[2]!=y[1]) return 0;}else if(C2!=0) return 1;}
else if(order==5){
  if(C5==Ca) {if(x_[5]!=x[1] || y_[5]!=y[1]) return 0;}else if(C5!=0) return 1;}
else if(order==6){
  if(C1==Ca) {if(x_[1]!=x[1] || y_[1]!=y[1]) return 0;}else if(C1!=0) return 1;}
}
else if(plc==7){
if(order==1){
mod_XY(4);
  if(C7==Ca) {if(x_[7]!=x[4] || y_[7]!=y[4]) return 0;}else if(C7!=0) return 1;}
else if(order==2){
  if(C3==Ca) {if(x_[3]!=x[4] || y_[3]!=y[4]) return 0;}else if(C3!=0) return 1;}
else if(order==3){
  if(C2==Ca) {if(x_[2]!=x[4] || y_[2]!=y[4]) return 0;}else if(C2!=0) return 1;}
```

```
else if(order==4){
  if(C5==Ca) {if(x_[5]!=x[4] || y_[5]!=y[4]) return 0;}else if(C5!=0) return 1;}
else if(order==5){
  if(C1==Ca) {if(x_[1]!=x[4] || y_[1]!=y[4]) return 0;}else if(C1!=0) return 1;}
else if(order==6){
  if(C4==Ca) {if(x_[4]!=x[4] || y_[4]!=y[4]) return 0;}else if(C4!=0) return 1;}
}
else if(plc==3){
if(order==1){
mod_XY(7);
  if(C3==Ca) {if(x_[3]!=x[7] || y_[3]!=y[7]) return 0;}else if(C3!=0) return 1;}
else if(order==2){
  if(C2==Ca) {if(x_[2]!=x[7] || y_[2]!=y[7]) return 0;}else if(C2!=0) return 1;}
else if(order==3){
  if(C5==Ca) {if(x_[5]!=x[7] || y_[5]!=y[7]) return 0;}else if(C5!=0) return 1;}
else if(order==4){
  if(C1==Ca) {if(x_[1]!=x[7] || y_[1]!=y[7]) return 0;}else if(C1!=0) return 1;}
else if(order==5){
  if(C4==Ca) {if(x_[4]!=x[7] || y_[4]!=y[7]) return 0;}else if(C4!=0) return 1;}
else if(order==6){
  if(C7==Ca) {if(x_[7]!=x[7] || y_[7]!=y[7]) return 0;}else if(C7!=0) return 1;}
}
else if(plc==2){
if(order==1){
mod_XY(3);
  if(C2==Ca) {if(x_[2]!=x[3] || y_[2]!=y[3]) return 0;}else if(C2!=0) return 1;}
else if(order==2){
  if(C5==Ca) {if(x_[5]!=x[3] || y_[5]!=y[3]) return 0;}else if(C5!=0) return 1;}
else if(order==3){
  if(C1==Ca) {if(x_[1]!=x[3] || y_[1]!=y[3]) return 0;}else if(C1!=0) return 1;}
else if(order==4){
  if(C4==Ca) {if(x_[4]!=x[3] || y_[4]!=y[3]) return 0;}else if(C4!=0) return 1;}
else if(order==5){
  if(C7==Ca) {if(x_[7]!=x[3] || y_[7]!=y[3]) return 0;}else if(C7!=0) return 1;}
else if(order==6){
  if(C3==Ca) {if(x_[3]!=x[3] || y_[3]!=y[3]) return 0;}else if(C3!=0) return 1;}
}
else if(plc==5){
if(order==1){
mod_XY(2);
  if(C5==Ca) {if(x_[5]!=x[2] || y_[5]!=y[2]) return 0;}else if(C5!=0) return 1;}
else if(order==2){
  if(C1==Ca) {if(x_[1]!=x[2] || y_[1]!=y[2]) return 0;}else if(C1!=0) return 1;}
else if(order==3){
  if(C4==Ca) {if(x_[4]!=x[2] || y_[4]!=y[2]) return 0;}else if(C4!=0) return 1;}
else if(order==4){
```

```
  if(C7==Ca) {if(x_[7]!=x[2] || y_[7]!=y[2]) return 0;}else if(C7!=0) return 1;}
else if(order==5){
  if(C3==Ca) {if(x_[3]!=x[2] || y_[3]!=y[2]) return 0;}else if(C3!=0) return 1;}
else if(order==6){
  if(C2==Ca) {if(x_[2]!=x[2] || y_[2]!=y[2]) return 0;}else if(C2!=0) return 1;}
}
}/**if(flag)**/
else{  /* for CCW */
if(plc==1){
if(order==1){
mod_XY(4);
  if(C1==Ca) {if(x_[1]!=x[4] || y_[1]!=y[4]) return 0;}else if(C1!=0) return 1;}
else if(order==2){
  if(C5==Ca) {if(x_[5]!=x[4] || y_[5]!=y[4]) return 0;}else if(C5!=0) return 1;}
else if(order==3){
  if(C2==Ca) {if(x_[2]!=x[4] || y_[2]!=y[4]) return 0;}else if(C2!=0) return 1;}
else if(order==4){
  if(C3==Ca) {if(x_[3]!=x[4] || y_[3]!=y[4]) return 0;}else if(C3!=0) return 1;}
else if(order==5){
  if(C7==Ca) {if(x_[7]!=x[4] || y_[7]!=y[4]) return 0;}else if(C7!=0) return 1;}
else if(order==6){
  if(C4==Ca) {if(x_[4]!=x[4] || y_[4]!=y[4]) return 0;}else if(C4!=0) return 1;}
}
else if(plc==5){
if(order==1){
mod_XY(1);
  if(C5==Ca) {if(x_[5]!=x[1] || y_[5]!=y[1]) return 0;}else if(C5!=0) return 1;}
else if(order==2){
  if(C2==Ca) {if(x_[2]!=x[1] || y_[2]!=y[1]) return 0;}else if(C2!=0) return 1;}
else if(order==3){
  if(C3==Ca) {if(x_[3]!=x[1] || y_[3]!=y[1]) return 0;}else if(C3!=0) return 1;}
else if(order==4){
  if(C7==Ca) {if(x_[7]!=x[1] || y_[7]!=y[1]) return 0;}else if(C7!=0) return 1;}
else if(order==5){
  if(C4==Ca) {if(x_[4]!=x[1] || y_[4]!=y[1]) return 0;}else if(C4!=0) return 1;}
else if(order==6){
  if(C1==Ca) {if(x_[1]!=x[1] || y_[1]!=y[1]) return 0;}else if(C1!=0) return 1;}
}
else if(plc==2){
if(order==1){
mod_XY(5);
  if(C2==Ca) {if(x_[2]!=x[5] || y_[2]!=y[5]) return 0;}else if(C2!=0) return 1;}
else if(order==2){
  if(C3==Ca) {if(x_[3]!=x[5] || y_[3]!=y[5]) return 0;}else if(C3!=0) return 1;}
else if(order==3){
  if(C7==Ca) {if(x_[7]!=x[5] || y_[7]!=y[5]) return 0;}else if(C7!=0) return 1;}
```

```
else if(order==4){
  if(C4==Ca) {if(x_[4]!=x[5] || y_[4]!=y[5]) return 0;}else if(C4!=0) return 1;}
else if(order==5){
  if(C1==Ca) {if(x_[1]!=x[5] || y_[1]!=y[5]) return 0;}else if(C1!=0) return 1;}
else if(order==6){
  if(C5==Ca) {if(x_[5]!=x[5] || y_[5]!=y[5]) return 0;}else if(C5!=0) return 1;}
}
else if(plc==3){
if(order==1){
mod_XY(2);
  if(C3==Ca) {if(x_[3]!=x[2] || y_[3]!=y[2]) return 0;}else if(C3!=0) return 1;}
else if(order==2){
  if(C7==Ca) {if(x_[7]!=x[2] || y_[7]!=y[2]) return 0;}else if(C7!=0) return 1;}
else if(order==3){
  if(C4==Ca) {if(x_[4]!=x[2] || y_[4]!=y[2]) return 0;}else if(C4!=0) return 1;}
else if(order==4){
  if(C1==Ca) {if(x_[1]!=x[2] || y_[1]!=y[2]) return 0;}else if(C1!=0) return 1;}
else if(order==5){
  if(C5==Ca) {if(x_[5]!=x[2] || y_[5]!=y[2]) return 0;}else if(C5!=0) return 1;}
else if(order==6){
  if(C2==Ca) {if(x_[2]!=x[2] || y_[2]!=y[2]) return 0;}else if(C2!=0) return 1;}
}
else if(plc==7){
if(order==1){
mod_XY(3);
  if(C7==Ca) {if(x_[7]!=x[3] || y_[7]!=y[3]) return 0;}else if(C7!=0) return 1;}
else if(order==2){
  if(C4==Ca) {if(x_[4]!=x[3] || y_[4]!=y[3]) return 0;}else if(C4!=0) return 1;}
else if(order==3){
  if(C1==Ca) {if(x_[1]!=x[3] || y_[1]!=y[3]) return 0;}else if(C1!=0) return 1;}
else if(order==4){
  if(C5==Ca) {if(x_[5]!=x[3] || y_[5]!=y[3]) return 0;}else if(C5!=0) return 1;}
else if(order==5){
  if(C2==Ca) {if(x_[2]!=x[3] || y_[2]!=y[3]) return 0;}else if(C2!=0) return 1;}
else if(order==6){
  if(C3==Ca) {if(x_[3]!=x[3] || y_[3]!=y[3]) return 0;}else if(C3!=0) return 1;}
}
else if(plc==4){
if(order==1){
mod_XY(7);
  if(C4==Ca) {if(x_[4]!=x[7] || y_[4]!=y[7]) return 0;}else if(C4!=0) return 1;}
else if(order==2){
  if(C1==Ca) {if(x_[1]!=x[7] || y_[1]!=y[7]) return 0;}else if(C1!=0) return 1;}
else if(order==3){
  if(C5==Ca) {if(x_[5]!=x[7] || y_[5]!=y[7]) return 0;}else if(C5!=0) return 1;}
else if(order==4){
```

```
  if(C2==Ca) {if(x_[2]!=x[7] || y_[2]!=y[7]) return 0;}else if(C2!=0) return 1;}
else if(order==5){
  if(C3==Ca) {if(x_[3]!=x[7] || y_[3]!=y[7]) return 0;}else if(C3!=0) return 1;}
else if(order==6){
  if(C7==Ca) {if(x_[7]!=x[7] || y_[7]!=y[7]) return 0;}else if(C7!=0) return 1;}
}
}/**else(flag)**/

order++;if(order==6) return 0;
}

return -1;
}/** nv **/



long get_len(int x1,int y1,int x2,int y2)
{
int dx,dy;

dx=x2-x1;
dy=y2-y1;

return /*sqrt*/(dx*dx+dy*dy-dx*dy);
}/** get_len **/


int check_len(int *nx,int *ny)
{
int i,j,k,m,dx,dy,lr,Li,Lj;
long len[3];

for(j=0;j<Lj;j++)
for(i=0;i<Li;i++){
dx=(RESO-1)*i;
dy=(RESO-1)*j;

/* left */
if(pixel[dx][dy]!=0 && pixel[dx+RESO-1][dy+RESO-1]!=0 && pixel[dx][dy+RESO-1]!=0){
for(k=0;k<CPMAX;k++){
if(nx[k]>=dx && ny[k]<=dy+RESO-1 && ny[k]>=nx[k]-dx+dy) ;
else goto right;
}

lr=0;
goto next;
}
```

```
/* right */
right:
if(pixel[dx][dy]!=0 && pixel[dx+RESO-1][dy+RESO-1]!=0 && pixel[dx+RESO-1][dy]!=0){
for(k=0;k<CPMAX;k++){
if(nx[k]<=dx+RESO-1 && ny[k]>=dy && ny[k]<=nx[k]-dx+dy) ;
else goto next_for;
}

lr=1;
goto next;
}

next_for:
;
}/**for(i)**/

return 0;

next:
for(k=0;k<CPMAX;k++){
if(k<CPMAX-1) m=k+1;else m=0;
len[k]=get_len(nx[k],ny[k],nx[m],ny[m]);
}

if(len[0]==len[1] && len[1]==len[2]){
/*printf(" i=%d j=%d\n",i,j);*/
if(RESO%3==1){
if(lr==0) {dx=RESO/3+(RESO-1)*i;          dy=RESO-(RESO/3+1)+(RESO-1)*j;}
else      {dx=RESO-(RESO/3+1)+(RESO-1)*i;dy=RESO/3+(RESO-1)*j;}

for(k=0;k<CPMAX;k++)
len[k]=get_len(dx,dy,nx[k],ny[k]);

if(len[0]==len[1] && len[1]==len[2]) return 1;
else return 2;
}/**if(RESO%3)**/
else return 1;
}/**if(len[]==len[])**/
else return 2;
}/** check_len **/



int cag_r(void)
{
int i,flag_us,dx,dy;
```

```
int flag_[CPMAX],flag_pp[CPMAX],acolor[6];
int nx[CPMAX],ny[CPMAX],nx_[CPMAX],ny_[CPMAX],nax[CPMAX],nay[CPMAX];
int XDP,YDP,nx_old[CPMAX],ny_old[CPMAX];
int cp,ssize;
int ca,c1,c2,c3,c4,c5,c7;
int nxp,nxm,nyp,nym;
int jmp[8],jp,tmp,dlt;
double val,angle;

if(0){
pixel[0][2*(RESO-1)]=-1;
pixel[xt][2*(RESO-1)]=-1;
}
/*if(RESO%3==1){
dx=RESO-1;dy=RESO-1;
XDP=RESO/3+dx;YDP=RESO-(RESO/3+1);pixel[XDP][YDP]=-1;
XDP=RESO-(RESO/3+1);YDP=RESO/3+dy*2;pixel[XDP][YDP]=-1;
}*/

ssize=sizeof(ss);
cp=CPMAX;

acolor[0]=9;acolor[1]=10;acolor[2]=11;
acolor[3]=7;acolor[4]=8;acolor[5]=12;

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

for(i=0;i<CPMAX;i++)
fp_mem[i]=0;

ca=15;
Ca=ca;

/*999*/
/* d1 */
nax[0]=2+RESO-1     ;nay[0]=4;
nax[1]=RESO-5+RESO-1;nay[1]=RESO-3;
nax[2]=2+RESO-1;nay[2]=RESO-3;

if(CPMAX==6){
nax[3]=4+RESO-1     ;nay[3]=2+_6d;
nax[4]=RESO-3+RESO-1;nay[4]=2+_6d;
nax[5]=RESO-3+RESO-1;nay[5]=RESO-5+_6d;
```

```
}

i=0;
while(1){
if(flag_[i]){                        /* CP_? */
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];
ig=i;

if(combination==0){
/* CW */
}/**if(combination)**/
else{
/* CCW */
if(drn==4) {/* 217 */
if(i<3){
     if(i%3==0) {nay[i]++;}
else if(i%3==1) {nax[i]--;nay[i]--;}
else if(i%3==2) {nax[i]++;}
}
else{
     if(i%3==0) {nax[i]++;nay[i]++;}
else if(i%3==1) {nax[i]--;}
else if(i%3==2) {nay[i]--;}
/*     if(i%3==0) {nax[i]++;}
else if(i%3==1) {nay[i]++;}
else if(i%3==2) {nax[i]--;nay[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)**/

/*use_subroop();*/
/****************************** while(cp) -> ******************************/

flag_us=0;

while(cp){                  /* 2pie/3 */
kbhit_();
if(refill==0) break;

algo=random_(2);

for(i=0;i<CPMAX;i++){
nx_old[i]=nx[i];
ny_old[i]=ny[i];
}


i=0;
while(1){

if(flag_[i]){                       /* CP_? */
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;jmp[1]=jmpflag;
c2=getpixel_(nx[i],ny[i],nx[i],nyp);
x[2]=X;y[2]=Y;jmp[2]=jmpflag;
c3=getpixel_(nx[i],ny[i],nxm,ny[i]);
x[3]=X;y[3]=Y;jmp[3]=jmpflag;
c4=getpixel_(nx[i],ny[i],nx[i],nym);
x[4]=X;y[4]=Y;jmp[4]=jmpflag;
c5=getpixel_(nx[i],ny[i],nxp,nyp);
x[5]=X;y[5]=Y;jmp[5]=jmpflag;
c7=getpixel_(nx[i],ny[i],nxm,nym);
x[7]=X;y[7]=Y;jmp[7]=jmpflag;

Nx=nx[i];Ny=ny[i];Nxp=nxp;Nyp=nyp;Nxm=nxm;Nym=nym;
C1=c1;C2=c2;C3=c3;C4=c4;C5=c5;C7=c7;

if((c1==ca)||(c2==ca)||(c3==ca)||(c4==ca)||(c5==ca)||(c7==ca)){
s.xx=nx[i];s.yy=ny[i];s.xx_=nx_[i];s.yy_=ny_[i];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 */



/*if(CPMAX==6){
if(i==3){
if(algo==0) algo=1;else algo=0;
}
}*/

if(algo==0){
val=-5;
if((/*c1!=ca*/nv(0,c1,1)==1)&&(c5==ca)){  /* CW (no phase) */
  if((angle=getangle(nxp,nyp))>val) {val=angle;nx[i]=x[5];ny[i]=y[5];jp=jmp[5];}}
if((/*c5!=ca*/nv(0,c5,5)==1)&&(c2==ca)){
  if((angle=getangle(std_x,nyp))>val) {val=angle;nx[i]=x[2];ny[i]=y[2];jp=jmp[2];}}
if((/*c2!=ca*/nv(0,c2,2)==1)&&(c3==ca)){
  if((angle=getangle(nxm,std_y))>val) {val=angle;nx[i]=x[3];ny[i]=y[3];jp=jmp[3];}}
if((/*c3!=ca*/nv(0,c3,3)==1)&&(c7==ca)){
  if((angle=getangle(nxm,nym))>val) {val=angle;nx[i]=x[7];ny[i]=y[7];jp=jmp[7];}}
if((/*c7!=ca*/nv(0,c7,7)==1)&&(c4==ca)){
  if((angle=getangle(std_x,nym))>val) {val=angle;nx[i]=x[4];ny[i]=y[4];jp=jmp[4];}}
if((/*c4!=ca*/nv(0,c4,4)==1)&&(c1==ca)){
  if((angle=getangle(nxp,std_y))>val) {val=angle;nx[i]=x[1];ny[i]=y[1];jp=jmp[1];}}
if(val==-5) /*beep(10000)*/;
}
else{
val=5;
if((/*c1!=ca*/nv(1,c1,1)==1)&&(c4==ca)){  /* CCW (no phase) */
  if((angle=getangle(std_x,nym))<val) {val=angle;nx[i]=x[4];ny[i]=y[4];jp=jmp[4];}}
if((/*c4!=ca*/nv(1,c4,4)==1)&&(c7==ca)){
  if((angle=getangle(nxm,nym))<val) {val=angle;nx[i]=x[7];ny[i]=y[7];jp=jmp[7];}}
if((/*c7!=ca*/nv(1,c7,7)==1)&&(c3==ca)){
  if((angle=getangle(nxm,std_y))<val) {val=angle;nx[i]=x[3];ny[i]=y[3];jp=jmp[3];}}
if((/*c3!=ca*/nv(1,c3,3)==1)&&(c2==ca)){
  if((angle=getangle(std_x,nyp))<val) {val=angle;nx[i]=x[2];ny[i]=y[2];jp=jmp[2];}}
if((/*c2!=ca*/nv(1,c2,2)==1)&&(c5==ca)){
  if((angle=getangle(nxp,nyp))<val) {val=angle;nx[i]=x[5];ny[i]=y[5];jp=jmp[5];}}
if((/*c5!=ca*/nv(1,c5,5)==1)&&(c1==ca)){
  if((angle=getangle(nxp,std_y))<val) {val=angle;nx[i]=x[1];ny[i]=y[1];jp=jmp[1];}}
if(val==5) /*beep(10000)*/;
}
```

```
if(jp==1){
nx_[i]=xt-nx_[i];
}
else if(jp==2){
tmp=nx_[i];nx_[i]=ny_[i];ny_[i]=tmp;
}
else if(jp==3){
tmp=nx_[i];nx_[i]=ny_[i];ny_[i]=tmp;
nx_[i]-=(_6dRow-1)*(RESO-1);ny_[i]+=(_6dRow-1)*(RESO-1);
}
else if(jp==4){
nx_[i]=4*(RESO-1)-nx_[i];ny_[i]=RESO-1;
}
else if(jp==5){
dlt=3*(RESO-1)-nx_[i];nx_[i]=RESO-1+dlt;ny_[i]=dlt;
}
else if(jp==6){
nx_[i]=2*(RESO-1)-nx_[i];ny_[i]=_6d;
}
else if(jp==7){
dlt=RESO-1-nx_[i];nx_[i]=RESO-1+dlt;ny_[i]=_6d+dlt;
}

if(1) 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_;
flag_pp[i]=0;
}/**else(c1,c2,c3,c4)**/
}/**if(flag_[i])**/

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

if(0){
i=0;
while(1){
ig=i;
if(flag_[i]==1 && flag_pp[i]==1) putpixel_(nx[i],ny[i],acolor[i]);

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

```
/*if(flag_us==0 && flag_[0]==1 && flag_pp[0]==1 && check_len(nx_old,ny_old)>1){
printf(" ?\n");
i=0;putpixel(nx_[i],ny_[i],12);
i=1;putpixel(nx_[i],ny_[i],12);
i=2;putpixel(nx_[i],ny_[i],12);
use_subroop();flag_us=1;
}*/
}/**while(cp)**/

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