EZDRAW

南台電子系　黎靖

一、EZDRAW

1. 安裝繪圖程式

Copy:　　“Ezd32m.lib”　to “Microsoft Visual Stdio\VC98\Lib\”
Copy:　　"EzDraw.ini"　to "c:\Windows\"
Copy:　　"EzDraw32.dll"　to "c:\Windows\System"
Copy: include 下所有的 .h 檔至 “c: Program files\Microsoft Visual Stdio\VC98\include

＊程式 down load: www.uop.edu/fordtopp or www.fordtopp.com or www.prenhall.com

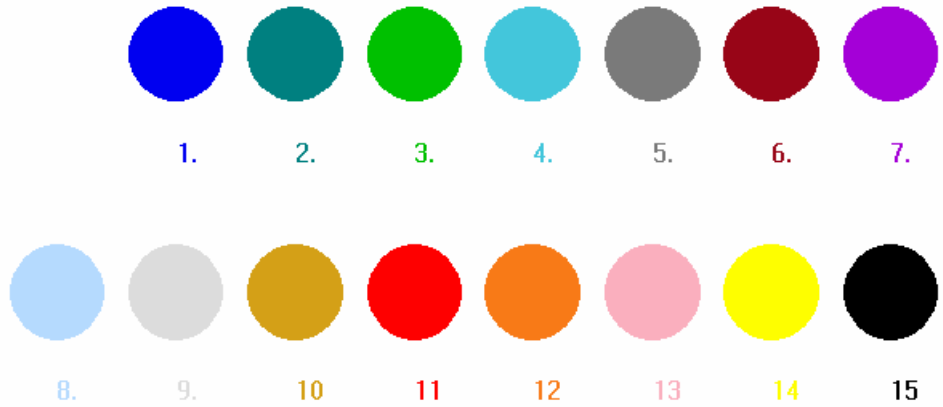
2. 下列函數存在 ezdraw.h

函數	說明
ezdInitialize(void)	開視窗
ezdClearDrawSurface()	清除視窗
ezdCleanUp();	關閉視窗
ezdWait((EZDUINT)(secs * 1000))	延遲視窗
keyPress()	輸入任意鍵
ezdDrawLine(x1, y1, x2, y2);	繪線
ezdDrawRectangle(x1, y1, x2, y2);	實心矩形
ezdDrawCircle(x, y, r)	實心圓形
ezdDrawPoint(x, y);	繪點
ezdDrawPolygon(n, Xarray, Yarray);	實心多邊形
ezdDrawText(Text, x, y);	寫上文字
ezdDeleteShape(EZDHANDLE hShape);	erase the shape
EZDCOLOR(r,g,b)	設定顏色
ezdSetColor(EZDCOLORVAL color)	設定顏色
ezdWaitForKeyPress()	等待按鍵輸入
ezdGetOriginY();	原點座標
ezdGetOriginY();	原點座標
ezdGetOriginType();	原點型態
ezdSetOriginType(EZDINT iMode);	設定原點型態
ezdSetOrigin(x, y, ezdOrigin Origin);	設定原點座標
ezdGetViewportWidth();	讀取視窗寬度
ezdGetViewportHeight();	讀取視窗高度

ezdSetViewport(w, h);	設定視窗大小
ezdGetMapMode();	讀取圖形模式
ezdSetMapMode(EZDINT iMode);	設定圖形模式
ezdGetForceAspect();	
ezdSetForceAspect(EZDBOOL bForce)	
ezdClearDrawSurface()	清除繪圖區域
ezdKeyPress();	按鍵偵測
ezdWait(EZDUINT uiMiliseconds);	等待一段時間

顏色的資料型態是 EZDCOLORVAL

預設的顏色：ezdWhite, ezdBlue, ezdTeal(青), ezdGreen, ezdTurquoise(青綠色), ezdDarkGray, ezdBrown, ezdPurple, ezdLightBlue, ezdLightGray, ezdGold, ezdRed, ezdOrange, ezdPink, ezdYellow, ezdBlack



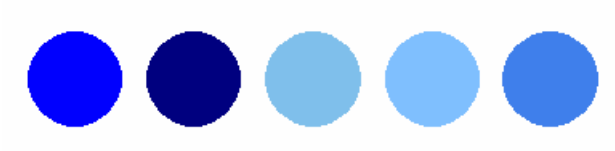

```
// 本程式顯示如何自行配色
#include "d_draw.h"
void main()
{
    openWindow();
    ezdSetColor(EZDCOLOR(0,255,0));    //Green
    ezdDrawCircle(1, 2, 0.4);
    viewWindow();    closeWindow();
}
```

白色系列	r	g	b
White	255	255	255
Ivory	255	255	243
Titanium white	255	255	247
Antique white	251	235	215

灰色系列	r	g	b
Light gray	191	191	191
Dark gray	127	127	127
Cold gray	115	127	127
Warm gray	127	127	107

黑色系列	r	g	b
Black	0	0	0
Ivory black	23	23	15
Lamp black	15	23	19

藍色系列	r	g	b
Blue	0	0	255
Navy blue	0	0	127
Sky blue	127	191	235
Sky blue (light)	127	191	255
Sky blue (dark)	63	127	235



黃色系列	r	g	b
Yellow	255	255	0
Light yellow	255	255	191
Gold	255	215	11
Cadmium yellow	255	191	15

綠色系列	r	g	b
Green	0	255	0
Dark green	0	127	0

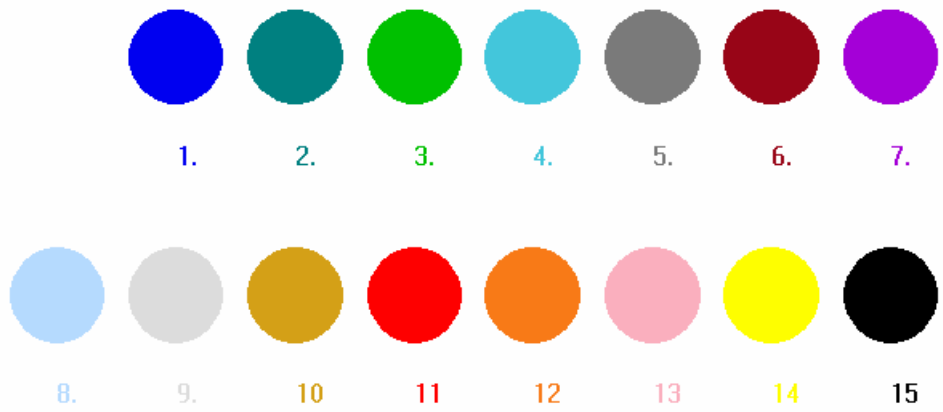
Olive green	79	127	63
Lime green	63	207	63

3. 下列函數存在 d_draw.h

函數	說明
openWindow()	開視窗 ezdInitialize()
viewWindow()	觀看視窗 ezdWaitForKeyPress()
closeWindow()	關閉視窗 ezdCleanUp();
eraseWindow()	清除視窗 ezdClearDrawSurface()
delayWindow(double secs)	延遲視窗 ezdWait((EZDUINT)(secs * 1000))
keyPress()	按鍵偵測 ezdKeyPress();

4. 下列函數存在 d_shape.h

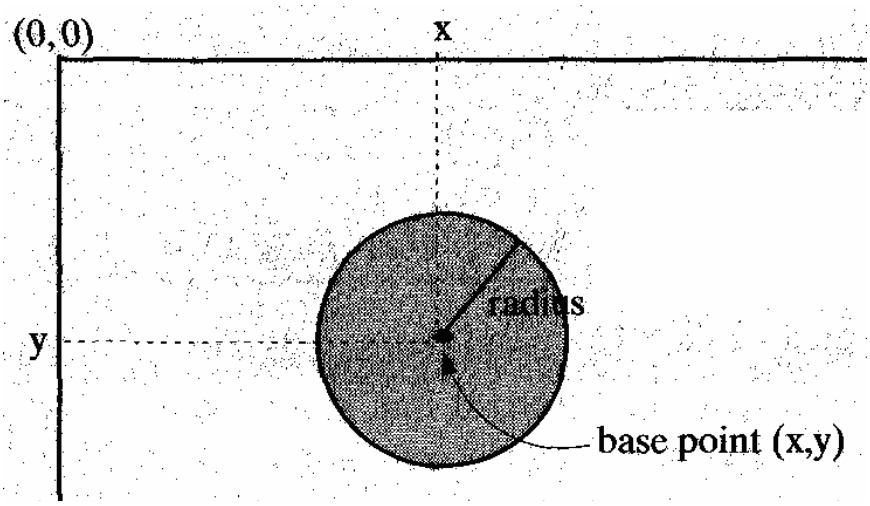
函數	說明
convertToEzdColor()	轉為 EZ Draw 預設之顏色名稱
getX(), getY()	輸出基點的座標
move(double x, double y)	基點的座標移至 (x,y)
getColor()	讀取物件的顏色
drawing_color(color)	設定繪圖的顏色
setColor(shapeColor c);	設定物件的顏色
erase()	刪除物件
update()	重畫物件
顏色的資料型態是 shapeColor 使用 d_shape.h 只能用預設的顏色：white, blue, teal(青), green, turquoise(青綠色), darkgray, brown, purple, lightblue, lightgray, gold, red, orange, pink, yellow, black	




```
//color.cpp
#include "d_draw.h"
#include "d_circsh.h"
#include "d_textsh.h"
struct POSI { double x, y; };
void DrawNum(float x, float y, float NUM, unsigned n, shapeColor c);
void main()
{
	shapeColor c = white;
	circleShape Circ(1, 2, 0.4, white);
	Circ.setRadius(0.4);
	openWindow();
	for(int i = 0; i < 8; i++)
	{
		Circ.setColor(c);
		Circ.move(i+1, 2);
		Circ.draw();
		DrawNum(i+1, 2.7, i, 2, c);
		c++;
	}
	for(i = 8; i < 16; i++)
	{
		Circ.setColor(c);
		Circ.move(i-7, 4);
		Circ.draw();
		DrawNum(i-7, 4.7, i, 2, c);
		c++;
	}
	viewWindow();	closeWindow();
}
```

```
void DrawNum(float x, float y, float NUM, unsigned n, shapeColor c)
{    // NUM is the number sting of n digits to be output
       static char CHAR[80];
       textShape text(x, y, gcvt(NUM,n, CHAR), c);
       text.draw();
}
```

5. 下列函數存在 d_circsh.h"



函數	說明
getRadius()	Output radius
setRadius(double r)	Setting radius

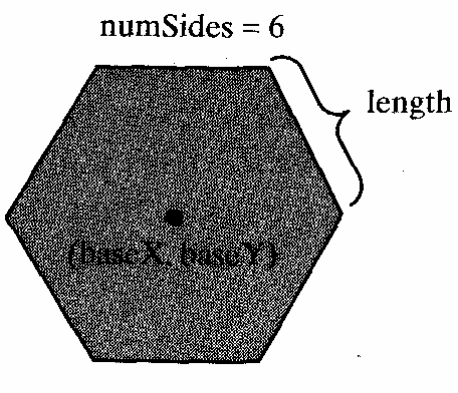
範例 1：定義及繪製實心圓物件

```
circleShape circle(VSx(P[i].x), VSy(P[i].y), radius, color);
circle.draw();
```

範例 2：

```
circleShape circ(5.0, 4.0, 2.0, lightgray);
circ.draw();        // draw the circle
circ.setRadius(1.0);    // set the radius as 1
circ.move(1.5, 4.0);      // reset the center of circ as (1.5, 4.0)
circ.setColor(darkgray);
circ.draw();

circleShape newCirc;
newX = circ.getX();      // newX = 1.5
newY = circgetY();      // newY = 4.0
newCirc.move(newX+3, newY-1);      // set the center at (4.5, 3.0)
newCirc.setRadius(2 * circ.getRadius());      // set the radius of newCirc as 2
circ.draw();

newCirc.draw();
```

6. 下列函數存在 d_polysh.h



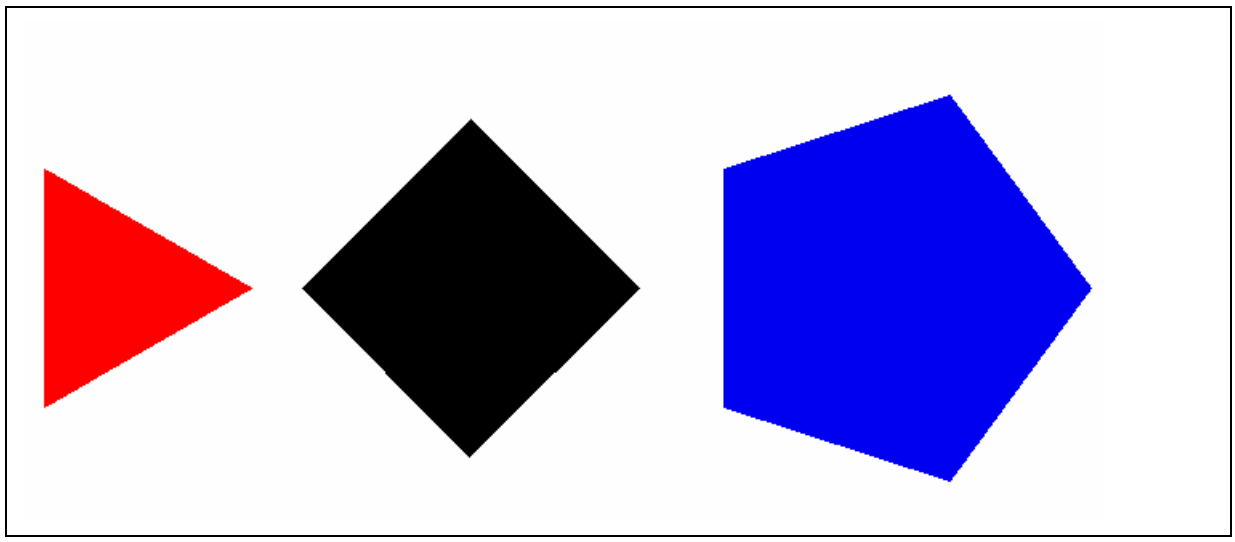
函數	說明
double getLength()	讀取邊長
setLength(double len)	設定邊長
getN()	讀取邊數
setN(int numsides)	設定邊數

範例 1：定義及繪製多邊形物件

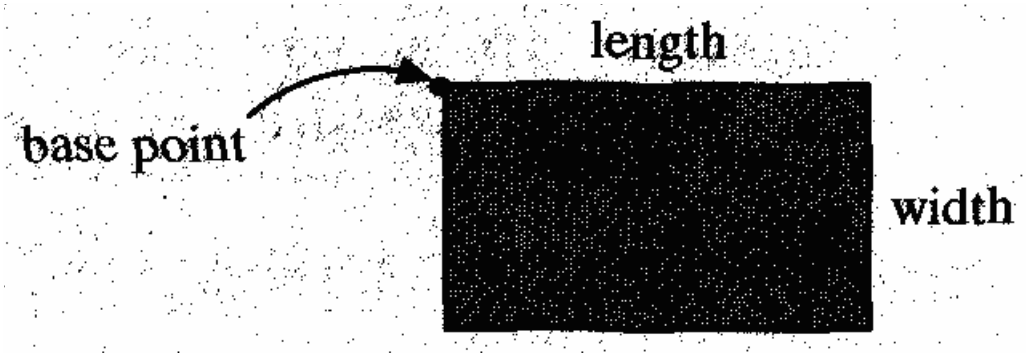
```
polyShape poly(VSx(P[i].x), VSy(P[i].y), n, length, color);
poly.draw();
```

範例 2：

```
// Triangle.cpp,
#include "d_draw.h"              // graphics library
#include "d_polysh.h"
void main()
{
    openWindow();
    polyShape poly1(1, 4, 3, 2, red);
    poly1.draw();
    polyShape poly2(4, 4, 4, 2, black);
    poly2.draw();
    polyShape poly3(7.5, 4, 5, 2, blue);
    poly3.draw();
    viewWindow();   closeWindow();
}
```



7. 下列函數存在 d_rectsh.h



函數	說明
setSides(double len, double wid)	設定邊長
getWidth()	讀取寬度
getLength()	讀取長度
perimeter()	讀取周長
area()	讀取面積

範例 1：定義及繪製矩形物件
rectShape square(3.0, 2.0, 4.0, 4.0, blue); // base.x = 3, base.y = 2, length = 4, width = 4
square.draw();	

8. 下列函數存在 d_linesh.h

函數	說明
double getEndX()	讀取終點座標 x
double getEndY()	讀取終點座標 y
void setEndPoint(double x, double y);	設定終點座標(x, y)
範例 1：定義及繪製直線 lineShape Line(x1, y1, x2, y2, color); Line.draw( );	

範例 2：

```
void DashLine(float x1, float y1, float x2, float y2, int n)
{
      int i = 0;
      float dx = (x2-x1)/(2*n + 1), dy = (y2 – y1)/(2*n + 1);
      lineShape Line(0, 0, 0, 0, black)
      Line.move(x1, y1);
      do   {
            x1 += dx;   y1 += dy;
            Line.setEndPoint(x1, y1);
            Line.draw();
            x1 += dx;   y1 += dy;
            Line.move(x1, y1);
      } while(++i <= n);
}
```

8. 下列函數存在 d_textsh.h

函數	說明
string getText()	取出字串
void setText(const string& s);	設定字串

範例：定義及輸出字串

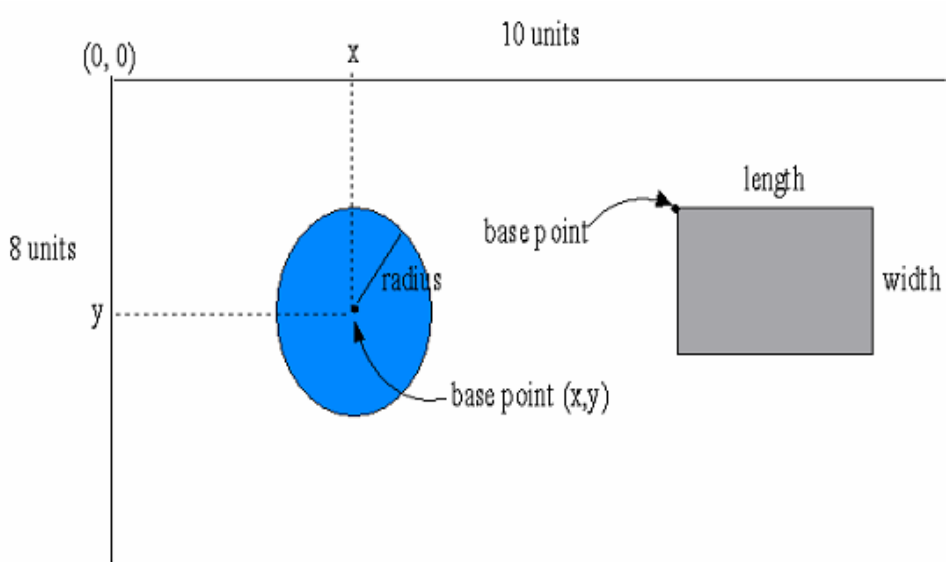
```
      textShape textA(3, 4, "A", red);   textA.draw();
```

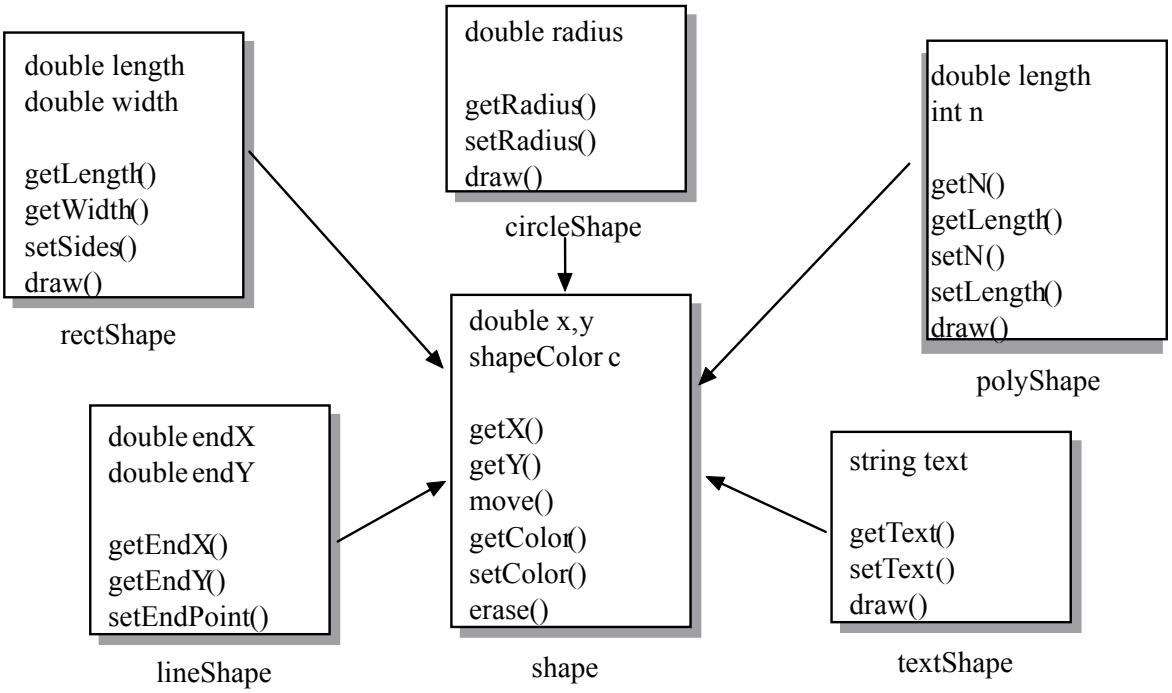
範例：輸出字串

```
void DrawText(float x, float y, char* s, shapeColor c)
{      textShape text(x, y, s, c);      text.draw();      }
```

範例：輸出數字

```
void DrawNum(float x, float y, float NUM, unsigned n, shapeColor c)
{   // NUM is the number sting of n digits to be output
      static char CHAR[80];
      textShape text(x, y, gcvt(NUM,n, CHAR), c);
      text.draw();
}
```

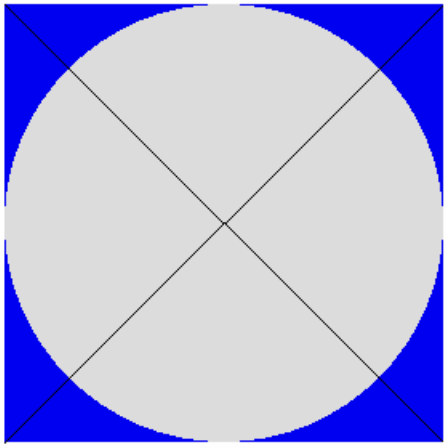

(0, 0)
x
10 units
8 units
y
radius
base point
length
width
base point (x,y)


double length
double width
getLength()
getWidth()
setSides()
draw()
rectShape
double radius
getRadius()
setRadius()
draw()
circleShape
double length
int n
getN()
getLength()
setN()
setLength()
draw()
polyShape
double x,y
shapeColor c
getX()
getY()
move()
getColor()
setColor()
erase()
shape
double endX
double endY
getEndX()
getEndY()
setEndPoint()
lineShape
string text
getText()
setText()
draw()
textShape

9. 範例

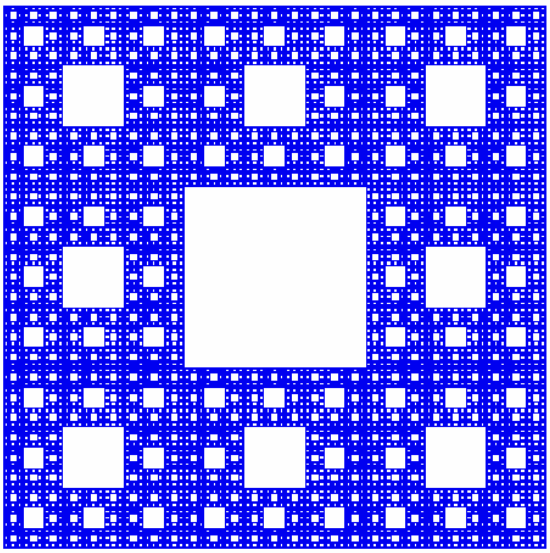
```
// File: prg13_2.cpp
// draw a 4 x 4 blue square centered at (5,4) in the drawing window.
// draw with a light gray circle with radius .125 centered at (5,4).
// after .1 seconds, draw the circle with the same center having a
// radius of .125 + .125. after pausing .1 seconds, continue this
// process with a circle whose radius increase by .125 until it
// reaches a radius of 2. at this point, the circle is inscribed in
// the square. draw the diagonals of the square and output the message
// "That's All Folks" below the square

#include "d_draw.h"       // graphics library
#include "d_rectsh.h"     // rectShape class
#include "d_circsh.h"     // circleShape class
#include "d_linesh.h"     // lineShape class
#include "d_textsh.h"     // textShape class

int main()
{
      // the light blue 4 x 4 square
      rectShape square(3.0, 2.0, 4.0, 4.0, blue);
      // circle that grows to size of the square
      circleShape circ(5.0, 4.0, 0, lightgray);
      // diagonal lines in the square
      lineShape diag1(3.0,2.0,7.0,6.0,black), diag2(3.0,6.0,7.0,2.0,black);
      // display message after the circle hits the bulls-eye
      textShape text(4.1,6.4,"That's All Folks", black);
      double r = 0.125;      // initial radius of the circle
      openWindow();       // open the drawing window and draw the square
      square.draw();
      do
      {
            circ.setRadius(r);       // set the radius and draw the circle
            circ.draw();
            r += 0.125;              // increase radius by 0.125
            delayWindow(.1);      // pause 1/10 second and then proceed
      } while (r <= 2.0);
      diag1.draw();  diag2.draw();      // draw the diagonals for the square
      text.draw();           // draw the label "That's All Folks"
```

```
    viewWindow();     // pause to view the final figure
    closeWindow();    // shutdown the drawing system
    return 0;
}
```




```
// Sierpinski.cpp
#include <math.h>
#include "d_draw.h"             // openWindow(); viewWindow();   closeWindow();
#include "d_rectsh.h"    // lineShape class

struct POSI { double x, y; };
void Sierpinski(int order, POSI base, float length);

void main()
{
    int order = 5;
    POSI base = {1, 1};
    float length=6;
    openWindow();
    rectShape square(base.x, base.y, length, length, blue);
    square.draw();
    delayWindow(0.2);

    Sierpinski(order, base, length);
    viewWindow();   closeWindow();
}

void Sierpinski(int order, POSI base, float length)
{
```

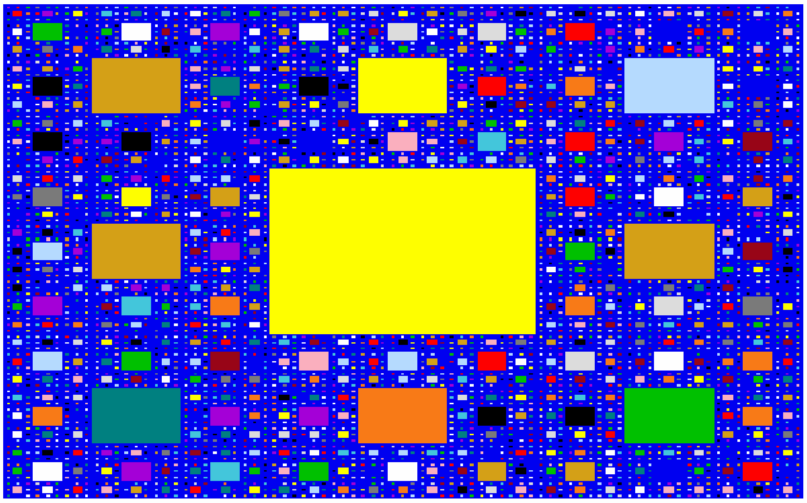
```
    POSI P[8];
    P[0] = base;
    if (0 == order)    return ;
    length /= 3.;
    base.x += length,    base.y += length;
    rectShape square(base.x, base.y, length, length, white);
    square.draw();
//  delayWindow(0.2);
    P[2].x = P[1].x = P[0].x;
    P[7].x = P[3].x = P[0].x + length;
    P[6].x = P[5].x = P[4].x = P[3].x + length;
    P[7].y = P[6].y = P[0].y;
    P[5].y = P[1].y = P[0].y + length;
    P[4].y = P[3].y = P[2].y = P[1].y + length;
    for (int i=0; i < 8; i++)    Sierpinski(order-1, P[i], length);
}
```



```
// Sierpinski.cpp
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <time.h>
#include "d_draw.h"              // openWindow(); viewWindow();    closeWindow();
#include "d_rectsh.h"     // lineShape class
#define ROUND(x)            ((x)>0 ? (int)((x)+0.5): -(int)(0.5-(x)))
struct POSI { double x, y; };
void Sierpinski(int order, POSI base, float length, float width);
int RandomInt(int lower, int upper);
drawing_color ValueToColor(float value, float min, float max);
```

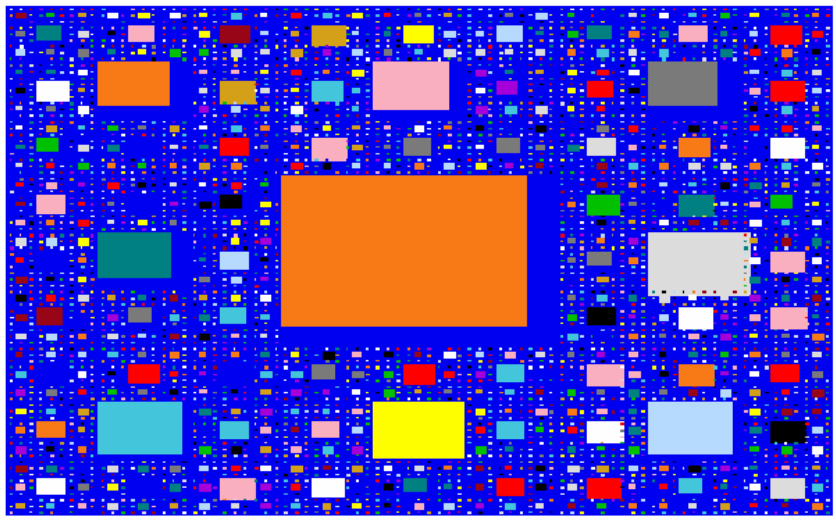
```
void main()
{
	int order = 5;
	POSI base = {1, 1};
	float length=8;
	float width = 0.618*length;
	srand(time(NULL)); /* use a random seed */
	openWindow();
	rectShape square(base.x, base.y, length, width, blue);
	square.draw();
	delayWindow(0.2);
	Sierpinski(order, base, length, width);
	viewWindow();    closeWindow();
}

void Sierpinski(int order, POSI base, float length, float width)
{
	unsigned c1 = RandomInt(0, 255);
	shapeColor color= ValueToColor(c1, 0, 16);
	POSI P[8];
	P[0] = base;
	if (0 == order)    return ;
	length /= 3.,    width /= 3.;
	base.x += length,    base.y += width;
	rectShape square(base.x, base.y, length, width, color);
	square.draw();
//	delayWindow(0.2);
	P[2].x = P[1].x = P[0].x;
	P[7].x = P[3].x = P[0].x + length;
	P[6].x = P[5].x = P[4].x = P[3].x + length;
	P[7].y = P[6].y = P[0].y;
	P[5].y = P[1].y = P[0].y + width;
	P[4].y = P[3].y = P[2].y = P[1].y + width;
	for (int i=0; i < 8; i++)    Sierpinski(order-1, P[i], length, width);
}

int RandomInt(int low, int high)
{
	if (low > high)        printf("RandomInt: low cannot be greater than high.");
	return (high - low + 1) * ((double) rand() / RAND_MAX) + low;
```

```
}

drawing_color ValueToColor(float value, float min, float max)
{
	unsigned color = 15-ROUND((value-min)*15/(max-min))%16;
	return(drawing_color(color));
}
```



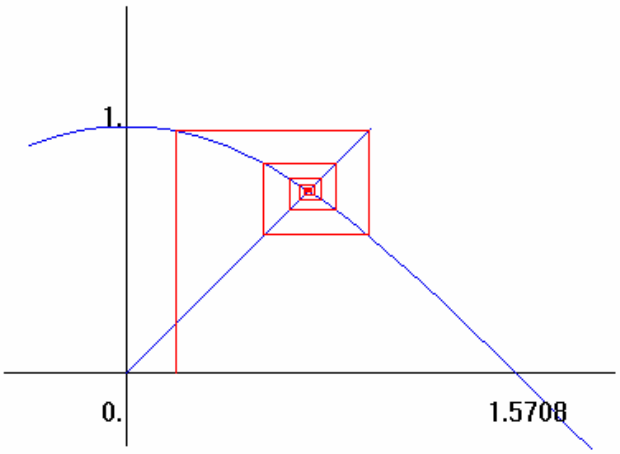
```
// Sierpinski.cpp
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <time.h>
#include "d_draw.h"		// openWindow(); viewWindow();	closeWindow();
#include "d_rectsh.h"	// lineShape class
#define ROUND(x)		((x)>0 ? (int)((x)+0.5): -(int)(0.5-(x)))
struct POSI { double x, y; };
void Sierpinski(int order, POSI base, float length, float width);
int RandomInt(int lower, int upper);
drawing_color ValueToColor(float value, float min, float max);

void main()
{
	int order = 5;
	POSI base = {1, 1};
	float length=8;
	float width = 0.618*length;
	srand(time(NULL)); /* use a random seed */
	openWindow();
	rectShape square(base.x, base.y, length, width, blue);
```

```
	square.draw();
	delayWindow(0.2);

	Sierpinski(order, base, length, width);
	viewWindow();	closeWindow();
}

void Sierpinski(int order, POSI base, float length, float width)
{
	unsigned c1 = RandomInt(0, 255);
	float f = 0.5+ RandomInt(20, 70)/100.;
	shapeColor color= ValueToColor(c1, 0, 16);
	POSI P[8];
	P[0] = base;
	if (0 == order)	return ;
	length /= 3.,	width /= 3.;
	base.x += length,	base.y += width;
	rectShape square(base.x, base.y, length*f, width*f, color);
	square.draw();
//	delayWindow(0.2);
	P[2].x = P[1].x = P[0].x;
	P[7].x = P[3].x = P[0].x + length;
	P[6].x = P[5].x = P[4].x = P[3].x + length;
	P[7].y = P[6].y = P[0].y;
	P[5].y = P[1].y = P[0].y + width;
	P[4].y = P[3].y = P[2].y = P[1].y + width;
	for (int i=0; i < 8; i++)	Sierpinski(order-1, P[i], length, width);
}

int RandomInt(int low, int high)
{
	if (low > high)	printf("RandomInt: low cannot be greater than high.");
	return (high - low + 1) * ((double) rand() / RAND_MAX) + low;
}

drawing_color ValueToColor(float value, float min, float max)
{
	unsigned color = 15-ROUND((value-min)*15/(max-min))%16;
	return(drawing_color(color));
}
```



使用迭代法求解:　x = cos(x)

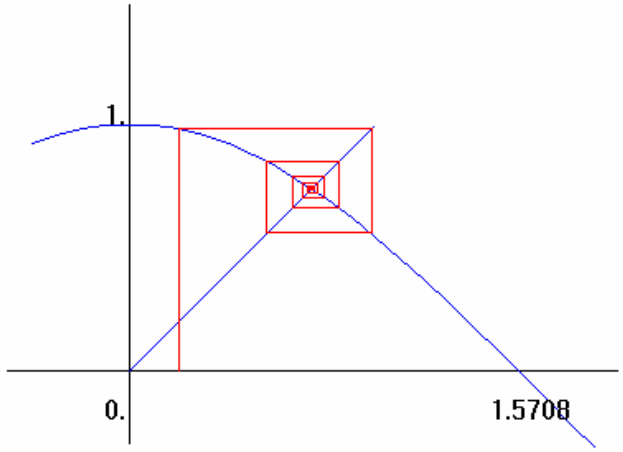
Solve x = cos(x)



i	x	cosx
0.	0.2	0.98007
1.	0.98007	0.55697
2.	0.55697	0.84886
3.	0.84886	0.66084
4.	0.66084	0.78948
5.	0.78948	0.70422
6.	0.70422	0.76212
7.	0.76212	0.72337
8.	0.72337	0.74958
9.	0.74958	0.73198
10	0.73198	0.74385
11	0.74385	0.73586
12	0.73586	0.74125
13	0.74125	0.73762
14	0.73762	0.74007
15	0.74007	0.73842

使用迭代法求解:　x = cos(x)

Solve x = cos(x)



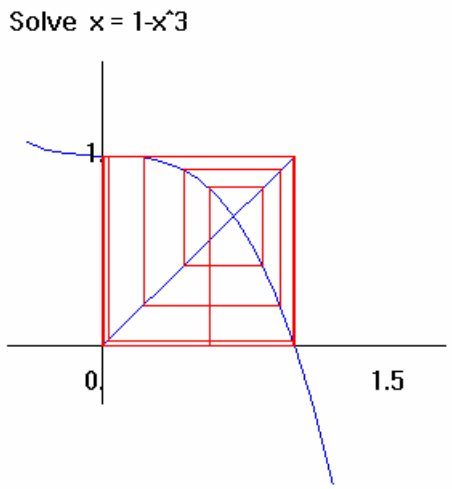
i	x	cosx
0.	0.2	0.98007
1.	0.98007	0.55697
2.	0.55697	0.84886
3.	0.84886	0.66084
4.	0.66084	0.78948
5.	0.78948	0.70422
6.	0.70422	0.76212
7.	0.76212	0.72337
8.	0.72337	0.74958
9.	0.74958	0.73198
10	0.73198	0.74385
11	0.74385	0.73586
12	0.73586	0.74125
13	0.74125	0.73762
14	0.73762	0.74007
15	0.74007	0.73842

```
// convergent.cpp
// Solve x = cos(x)
#include <stdio.h>
#include <math.h>
#include "d_draw.h"       // graphics library, openWindow(); viewWindow();    closeWindow();
#include "d_linesh.h"     // lineShape class
#include "d_textsh.h"
struct WorkingSpace { float x1, y1, x2, y2; } WS;
struct POSI { double x, y; };

void      setWindow(float, float, float, float);
void DrawLine(POSI p1, POSI p2, shapeColor c);
void DrawNum(float x, float y, float NUM, unsigned n, shapeColor c);
float Root(float x);
void DrawCos();

#define VSx(x)      (((x)-WS.x1)*mfx)
#define VSy(y)      ((WS.y2-(y))*mfy)
```

```
#define PI (3.14159)
float mfx, mfy;

void main()
{
	float x;
	printf("Solve x = cos(x).\nEnter initial guess\n");
	scanf("%f", &x);
	setWindow(-0.7,-1, 3, 3);
	textShape text(VSx(-0.5), VSy(2.8), "Solve    x = cos(x)", black);
	text.draw();
	DrawCos();
	system("pause");
	x = Root(x);
	viewWindow();    closeWindow();
}

void DrawCos()
{
	float x;
	POSI p1={-0.5, 0}, p2={2.0, 0};
	DrawLine(p1, p2, black);    // x axis
	p1.x = 0, p1.y = -0.3;
	p2.x = 0, p2.y = 1.5;
	DrawLine(p1, p2, black);      // y axis
	p1.x = -0.4, p1.y = cos(p1.x);
	for (x=-0.3; x <= 2; x+= 0.1)    // draw y = cos(x)
	{
		p2.x = x, p2.y = cos(x);
		DrawLine(p1, p2, blue);
		p1 = p2;
	}
	p1.x = p1.y = 0;
	p2.x = p2.y = 1;
	DrawLine(p1, p2, blue);      // draw y = x
	DrawNum(-0.1, -0.1, 0, 2, black);
	DrawNum(PI/2.-0.1, -0.1, PI/2, 5, black);
	DrawNum(-0.1, 1.1, 1, 2, black);
}
```

```
float Root(float x)
{
	POSI p1, p2;
	int i=0;
	float x_new;
	textShape text(VSx(2.5), VSy(2.8), "i                x                cosx", black);
	text.draw();
	p1.x = x, p1.y = 0;
	do {
		x_new = cos(x);
		DrawNum(2.5, 2.8-0.2*(i+1), i, 2, black);
		DrawNum(2.8, 2.8-0.2*(i+1), x, 5, black);
		DrawNum(3.4, 2.8-0.2*(i+1), x_new, 5, black);

		p2.x = p1.x, p2.y = x_new;
		DrawLine(p1, p2, red); p1 = p2;
		printf("%f\n", x_new);
		if(fabs(x_new-x) < 1e-5) return x_new;
		x = x_new;
		p2.x = x_new;
		DrawLine(p1, p2, red); p1 = p2;
		system("pause");
	} while(++i < 16);
	return x_new;
}

void setWindow(float x1, float y1, float x2, float y2)
{
	WS.x1 = x1; WS.y1 = y1; WS.x2 = x2; WS.y2 = y2;
	openWindow();
	mfx = 10./(WS.x2-WS.x1);    mfy = 8./(WS.y2-WS.y1);
	mfx = mfy = (mfx < mfy ? mfx : mfy);
}

void DrawLine(POSI p1, POSI p2, shapeColor c)
{
	lineShape Line(VSx(p1.x), VSy(p1.y),VSx(p2.x), VSy(p2.y), c) ;
	Line.draw();
}
```

```
void DrawNum(float x, float y, float NUM, unsigned n, shapeColor c)
{    // NUM is the number sting of n digits to be output
       static char CHAR[80];
       textShape text(VSx(x), VSy(y), gcvt(NUM,n, CHAR), c);
       text.draw();
}
```



i	x	1-x^3
0.	0.55	0.83363
1.	0.83363	0.42069
2.	0.42069	0.92555
3.	0.92555	0.20714
4.	0.20714	0.99111
5.	0.99111	2.6428e-002
6.	2.6428e-002	0.99998
7.	0.99998	5.5431e-005
8.	5.5431e-005	1.
9.	1.	0.
10	0.	1.
11	1.	0.

```
// divergent.cpp
#include <stdio.h>
#include <math.h>
#include "d_draw.h"       // graphics library, openWindow(); viewWindow();    closeWindow();
#include "d_linesh.h"     // lineShape class
#include "d_textsh.h"
struct WorkingSpace { float x1, y1, x2, y2; } WS;
struct POSI { double x, y; };

void      setWindow(float, float, float, float);
void DrawLine(POSI p1, POSI p2, shapeColor c);
void DrawNum(float x, float y, float NUM, unsigned n, shapeColor c);
float Root(float x);
void DrawFn();

#define VSx(x)      (((x)-WS.x1)*mfx)
#define VSy(y)      ((WS.y2-(y))*mfy)
#define PI (3.14159)
#define Cubic(x)    ((x)*(x)*(x))
float mfx, mfy;

void main()
```

```
{
	float x;
	printf("Enter initial guess\n");
	scanf("%f", &x);
	setWindow(-0.7,-2, 3, 3);
	textShape text(VSx(-0.5), VSy(2.8), "Solve    x = 1-x^3", black);
	text.draw();
	DrawFn();
	system("pause");
	x = Root(x);
	viewWindow();    closeWindow();
}

void DrawFn()
{
	float x;
	POSI p1={-0.5, 0}, p2={1.8, 0};
	DrawLine(p1, p2, black);        // x axis
	p1.x = 0, p1.y = -0.3;
	p2.x = 0, p2.y = 1.5;
	DrawLine(p1, p2, black);        // y axis
	p1.x = -0.4, p1.y = 1 - Cubic(p1.x);
	for (x=-0.3; x <= 1.3; x+= 0.1)        // draw y = 1-x^3
	{
		p2.x = x, p2.y = 1. - Cubic(x);
		DrawLine(p1, p2, blue);
		p1 = p2;
	}
	p1.x = p1.y = 0;
	p2.x = p2.y = 1;
	DrawLine(p1, p2, blue);        // draw y = x
	DrawNum(-0.1, -0.1, 0, 2, black);
	DrawNum(1.5-0.1, -0.1, 1.5, 5, black);
	DrawNum(-0.1, 1.1, 1, 2, black);
}

float Root(float x)
{
	POSI p1, p2;
	int i=0;
```

```
	float x_new;
	textShape  text(VSx(2.3),  VSy(2.8),  "i                    x                                1-x^3",
black);
	text.draw();
	p1.x = x, p1.y = 0;
	do {
		x_new = 1. - Cubic(x);
		DrawNum(2.3, 2.8-0.2*(i+1), i, 2, black);
		DrawNum(2.5, 2.8-0.2*(i+1), x, 5, black);
		DrawNum(3.6, 2.8-0.2*(i+1), x_new, 5, black);

		p2.x = p1.x, p2.y = x_new;
		DrawLine(p1, p2, red); p1 = p2;
		printf("%f\n", x_new);
		if(fabs(x_new-x) < 1e-5) return x_new;
		x = x_new;
		p2.x = x_new;
		DrawLine(p1, p2, red); p1 = p2;
		system("pause");
	} while(++i < 12);
	return x_new;
}

void setWindow(float x1, float y1, float x2, float y2)
{
	WS.x1 = x1; WS.y1 = y1; WS.x2 = x2; WS.y2 = y2;
	openWindow();
	mfx = 10./(WS.x2-WS.x1);    mfy = 8./(WS.y2-WS.y1);
	mfx = mfy = (mfx < mfy ? mfx : mfy);
}

void DrawLine(POSI p1, POSI p2, shapeColor c)
{
	lineShape Line(VSx(p1.x), VSy(p1.y),VSx(p2.x), VSy(p2.y), c) ;
	Line.draw();
}

void DrawNum(float x, float y, float NUM, unsigned n, shapeColor c)
{   // NUM is the number sting of n digits to be output
	static char CHAR[80];
```

```
    textShape text(VSx(x), VSy(y), gcvt(NUM,n, CHAR), c);
    text.draw();
}
```

使用下列公式求π

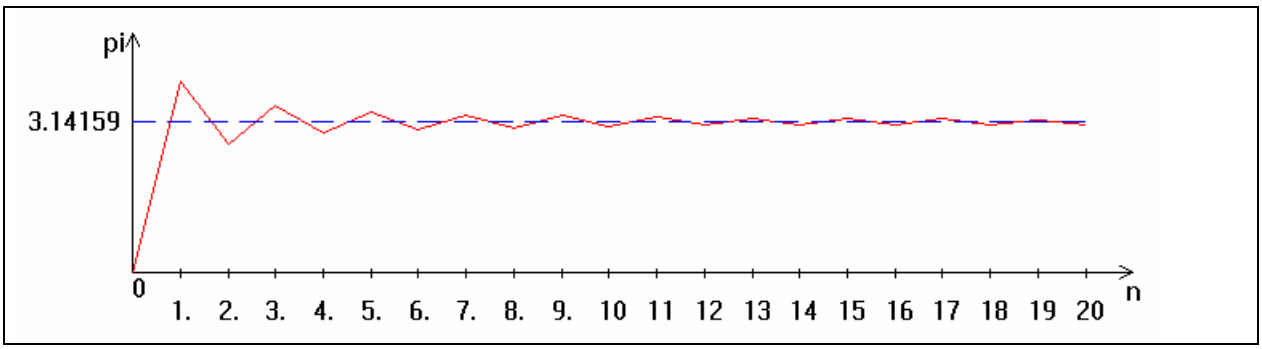
$$\pi = 4 \times \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} \pm \ldots\right) = 4 \times \sum_{n=1}^{\infty} \left((-1)^{n-1} \frac{1}{2n-1}\right)$$

```
// pi1.cpp
#include <stdio.h>
#include <math.h>
#include "d_shape.h"      // include d_draw.h
    // graphics library, ezdraw.h , openWindow(); viewWindow();    closeWindow(), ;
#include "d_linesh.h"            // lineShape class
#include "d_textsh.h"
#include "d_circsh.h"

struct WorkingSpace { float x1, y1, x2, y2; } WS;
struct POSI { double x, y; } ;

void DrawCoordinate(const POSI*, int, shapeColor c);
void Axis_x(float L, float R,    char *ch, shapeColor c);
void Axis_y(float L, float R,    char *ch, shapeColor c);
void DashLine(POSI star, POSI end, int n, shapeColor c) ;
void DrawLine(POSI p1, POSI p2, shapeColor c);
void setWindow(float x1, float y1, float x2, float y2);
float mfx, mfy;
#define VSx(x)      (((x)-WS.x1)*mfx)
#define VSy(y)      ((WS.y2-(y))*mfy)

void main()
{
    float pi4 = 0.;    // pi4 = PI/4
    unsigned n = 0;
    float sgn;
    POSI P[21];
    P[0].x = P[0].y = 0.;
    static char CHAR[80];
```

```
    while (fabs(pi4*4-3.14159)> 1e-6F)
    {
        n++;
        sgn = (n & 1? 1.: -1.);
        pi4 += (sgn/(2.*n-1.));
        if (n < 21)
        {
            P[n].x = n;
            P[n].y = pi4*4.;
        }
    }
    printf("n = %d PI = %f\n", n, pi4*4.);

    setWindow(-3, -3, 22, 8);    // x1, y1, x2, y2
    for(n = 0; n < 20; n++)          DrawLine(P[n], P[n+1], red);

    DrawCoordinate(P, 21, black);

    POSI p0 = {0, 3.14}, p1 = {20, 3.14};
    DashLine(p0, p1, 20, blue);
    textShape text0(VSx(0-2.2), VSy(3.5),gcvt(3.14159,7,CHAR),black);
    text0.draw();
    for(n = 1; n <= 20; n++)
    {
        p0.x = n, p0.y = -0.1;
        p1.x = n, p1.y = 0.1;
        DrawLine(p0, p1, black);
        textShape text0(VSx(n-0.2), VSy(-0.5),gcvt(n,2,CHAR),black);
        text0.draw();
    }
    viewWindow();      closeWindow();
}

void DrawCoordinate(const POSI* P, int n, shapeColor c)
{
    register unsigned i; float xmax=0, ymax=0;
    const float over=1;
    for (i=0; i < n; i++)
    {
        xmax = (P[i].x > xmax?    P[i].x: xmax);
```

```
            ymax = (P[i].y > ymax?    P[i].y: ymax);
        }
        xmax += over;    ymax += over;
        Axis_x(0, xmax, "n", c);
        Axis_y(0, ymax, "pi", c);
        textShape text0(VSx(0), VSy(0),"0",black);
        text0.draw();
}

void Axis_x(float L, float R,    char *ch, shapeColor c)
{                                                          // L=座標起點, R=座標終點
        POSI p1, p2, base;      float eps = 0.06/mfx;
        // Draw axis
        p1.x = p1.y = p2.x = p2.y = 0;
        {
            p1.x = R;        p2.x = L;
            DrawLine(p1, p2, c); // Draw Axis
            p2.x = p1.x-2*eps;    p2.y = p1.y+eps;
            DrawLine(p1, p2, c);    // Draw Arrow
            p2.y = p1.y-eps; DrawLine(p1, p2, c);    // Draw Arrow
            base.x = p1.x-eps;    base.y = p1.y-0.5*eps;
            textShape text0(VSx(base.x), VSy(base.y),ch,black);
            text0.draw();
        }
}

void Axis_y(float L, float R,    char *ch, shapeColor c)
{                                                          // L=座標起點, R=座標終點
        POSI p1, p2, base;      float eps = 0.06/mfx;
        // Draw axis
        p1.x = p1.y = p2.x = p2.y = 0;
        {
            p1.y = R;        p2.y = L;
            DrawLine(p1, p2, c); // Draw Axis
            p2.x = p1.x-eps;    p2.y = p1.y-2*eps;
            DrawLine(p1, p2, c);    // Draw Arrow
            p2.x = p1.x+eps;    DrawLine(p1, p2, c);    // Draw Arrow
            base.x = p1.x-4*eps;    base.y = p1.y+eps;
            textShape text0(VSx(base.x), VSy(base.y),ch,c);
            text0.draw();
```

```
    }
}

void DashLine(POSI star, POSI end, int n, shapeColor c)
{
    int i = 0;
    lineShape Line; Line.setColor(c);
    star.x = VSx(star.x);
    star.y = VSy(star.y);
    end.x = VSx(end.x);
    end.y = VSy(end.y);

    float dx = (end.x-star.x)/(2*n + 1), dy = (end.y - star.y)/(2*n + 1);
    Line.move(star.x, star.y);
    do  {
        star.x += dx;   star.y += dy;
        Line.setEndPoint(star.x, star.y);
        Line.draw();
        star.x += dx;   star.y += dy;
        Line.move(star.x, star.y);
    } while(++i <= n);
}

void DrawLine(POSI p1, POSI p2, shapeColor c)
{
    lineShape Line(VSx(p1.x), VSy(p1.y),VSx(p2.x), VSy(p2.y), c) ;
    Line.draw();
}

void setWindow(float x1, float y1, float x2, float y2)
{
    WS.x1 = x1; WS.y1 = y1; WS.x2 = x2; WS.y2 = y2;
    openWindow();
    mfx = 10./(WS.x2-WS.x1);
    mfy = 8./(WS.y2-WS.y1);
    mfx = mfy = (mfx < mfy? mfx: mfy);
}
```



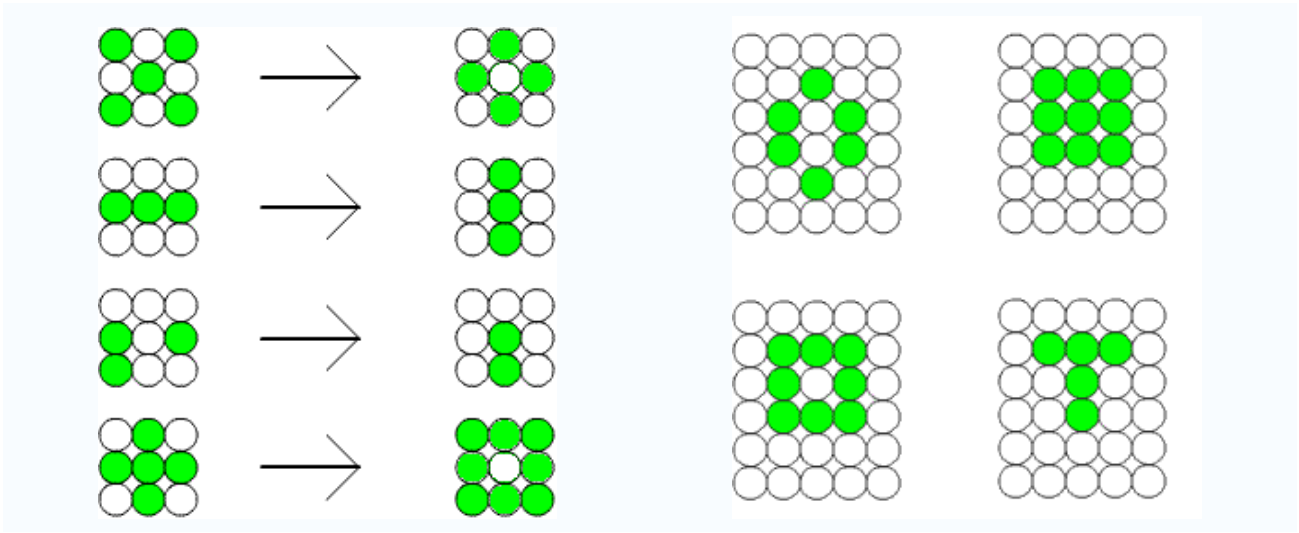
二、生命遊戲

生命遊戲是英國數學家約翰·何頓·康威在 1970 年發明的細胞自動機（也翻譯成「格狀自動機」）。它最初於 1970 年 10 月在《科學美國人》雜誌中馬丁·葛登能的「數學遊戲」專欄出現。

生命遊戲其實是一個零玩家遊戲。它包括一個二維矩形世界，這個世界中的每個方格居住著一個活著的或死了的細胞。某一細胞的鄰居包括上、下、左、右、左上、左下、右上與右下相鄰之細胞，一個細胞在下一個時刻生死取決於相鄰八個方格中活著的或死了的細胞的數量。遊戲規則如下：

1. 孤單死亡：如果細胞的鄰居小於一個，這個細胞會因太孤單而死去。
2. 擁擠死亡：如果細胞的鄰居在四個以上，這個細胞會因為資源匱乏而在下一個時刻死去。
3. 穩定：如果細胞的鄰居為二個或三個，則下一次狀態為穩定存活。
4. 復活：如果某位置原無細胞存活，而該位置的鄰居為三個，則該位置將復活一細胞。

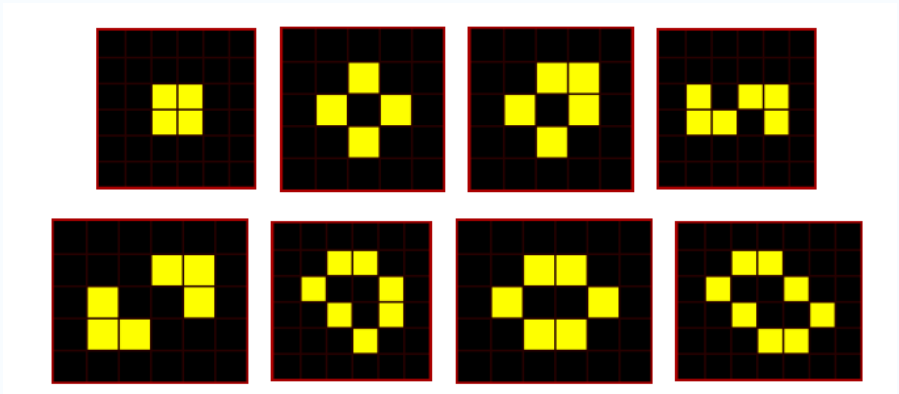
在這個遊戲中，還可以設定一些更加複雜的規則，例如當前方格的狀況不僅由父一代決定，而且還考慮祖父一代的情況。你還可以作為這個世界的 God，隨意設定某個方格細胞的死活，以觀察對世界的影響。



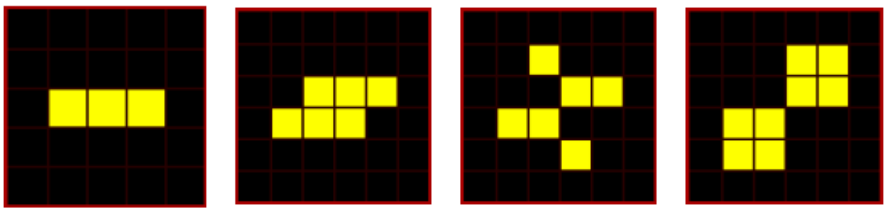
當 Conway 提出生命遊戲後，馬上造成轟動，不只是一些普通人在玩，而一些有名的數學家及電腦學家也樂此不疲。造成轟動的原因是，沒有人想到僅僅幾條簡單的規則，竟然就能產生類似於生命演化過程中無比複雜的現象。當你把這幾個簡單的規則變成電腦程式以後，它們似乎真的會讓電腦螢幕活起來。

用心注視，螢幕上活躍著各種動作，就好像你用顯微鏡觀察一滴池塘水所見到的微生物一樣。開始的時候，你可以讓螢幕上隨意散佈著活細胞，然後就會看到它們自我組織成各式各樣連貫性的結構。有的會滾動，有的好像野獸呼吸一般來回震盪，你還可以發現「滑翔機」——一群活細胞以固定的速度滑過螢幕，還有「滑翔機—機關槍」穩定的發射出新的滑翔機，以及其他結構不動聲色的把滑翔機一一吃掉。你或許還可看到一隻優閒的金魚，搖頭晃腦地上下擺動，然後消失在池塘邊。每一次出現的畫面都不一樣，沒有人看過所有可能的畫面。你將在下個單元看見這些範例 ，不過在看見那些景象之前，筆者必須先介紹 Conway 生命遊戲中簡單的穩定結構的幾個例子，我們可以將這些穩定的結構分成三類：

第一類：在疊代過程中，細胞群不會改變其狀態（形狀）。

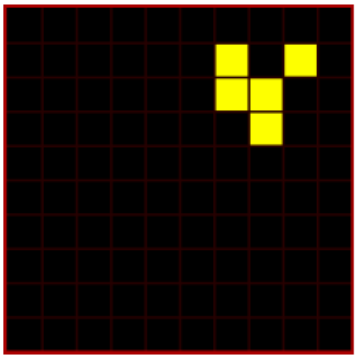


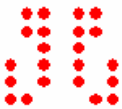
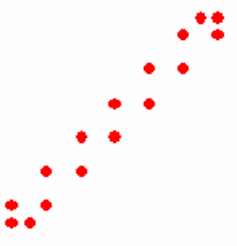
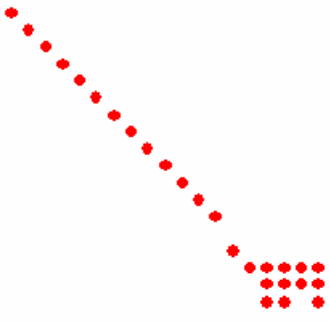
第二類：在疊代過程中，細胞群會在有限的疊代次數內週期循環其狀態（形狀），最著名的代表就是所謂的「閃光燈」（blinker）。

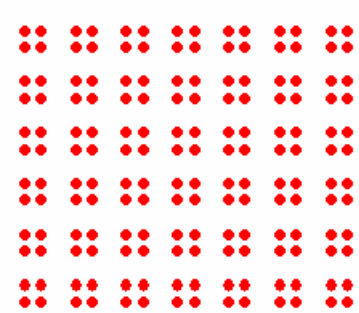


第三類：是第二類的延伸，除了會週期循環其狀態（形狀）之外，還會穩定地移動，最有名的代表就是所謂的「滑翔機」（glider）。

像滑翔機這種會移動並維持形態的結構，似乎能夠在生命遊戲中扮演傳遞訊號的角色 。能夠產生「滑翔機」的生命遊戲，代表著它具有訊號儲存與訊號傳遞的功能，而訊號（資料）的儲存與傳遞（像是 DNA ）是生物演化機制的重要特徵，也是建構一台電腦的必要條件，這符合 von Neumann 當初的看法：細胞自動機可以建構一個以人工生命為計算單位的電腦雛形。同樣地， Conway 也指出，飛行的滑翔機可以想像成川流不息的位元，滑翔機若能與其他滑翔機做巧妙的結合，便可以做符號運算，例如滑翔機的碰撞相當於邏輯閘，入射機群為輸入，而碰撞後的碎片為輸出， Conway 用數學去證明生命遊戲在邏輯上，足以涵蓋多用途數位元電腦的全部功能，他甚至認為細胞自動機可以成為一部萬用電腦。雖然 von Neumann 與 Conway 的理論構想是成立的，但是目前還沒有人實際做出來。



Glider Gun	Cheshire Cat	Tumbler	Barber Pole	Harvester
				
Glider Gun				

Virus	

```
// lifenew.cpp
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include "d_draw.h"       // graphics library, openWindow(); viewWindow();    closeWindow();
#include "d_linesh.h"     // lineShape class
#include "d_rectsh.h"
struct WorkingSpace { float x1, y1, x2, y2; } WS;
struct POSI { double x, y;} ;

#define     MAXROW              60              /* maximum row range            */
#define     MAXCOL              60              /*    maximum column range      */
#define VSx(x)     (((x)-WS.x1)*mfx)
#define VSy(y)     ((WS.y2-(y))*mfy)

void     CopyMap(bool map[][MAXCOL+2], bool newmap[][MAXCOL+2]);
void     Initialize(bool map[][MAXCOL+2], int);
int      NeighborCount(bool map[][MAXCOL+2], int row, int column);
void     WriteMap(bool map[][MAXCOL+2]);
void     setWindow(float, float, float, float);
double Rand_0_1(long int *seed);
void DrawLine(POSI p1, POSI p2, shapeColor c);
float mfx, mfy;
void main(void)
{
      long int seed;
      int    row, col,n, mode, N = 100;
      time(&seed);
      srand((unsigned int)(time(NULL)%10000));
      bool map[MAXROW+2][MAXCOL+2]={0}, newmap[MAXROW+2][MAXCOL+2]={0};
      printf("Choose an initial state:\n");
      printf("1 : Virus\n");
      printf("2 : Barber_Pole\n");
      printf("3 : blinker\n");
      printf("4 : blinker2\n");
      printf("5 : glider\n");
      printf("6 : Glider_Gun\n");
      printf("7 : Tumbler\n");
      printf("8 : Tumbler2\n");
      printf("9 : R_Pentomino\n");
```

```
    printf("10 : Cheshire_Cat\n");
    printf("11 : Harvester\n");
    printf("12 : Random\n");
    scanf("%d", &mode);
    if (mode < 12)        Initialize(map, mode);
    else
    {
        for (row = 1; row <= MAXROW; row++)
            for (col = 1; col <= MAXCOL; col++)
                if (Rand_0_1(&seed) < 0.2)    map[row][col]= 1;
    }
    setWindow(-5,-5,MAXCOL+5,MAXROW+5);
    rectShape square(VSx(0), VSy(61), 61.*mfy, 61.*mfx , lightblue);    square.draw();
    WriteMap(map);
    n = 0;
    do {
        delayWindow(1);
        for (row = 1; row <= MAXROW; row++)
            for (col = 1; col <= MAXCOL; col++)
            {
                switch(NeighborCount(map, row, col))
                {
                case 2:    newmap[row][col]= map[row][col];      break;
                case 3:    newmap[row][col]= 1;      break;
                default:    newmap[row][col]= 0;       break;
                }
            }
        CopyMap(map, newmap);
        WriteMap(map);
    } while (++n < N);
    viewWindow();    closeWindow();
}

int NeighborCount(bool map[][MAXCOL+2], int row, int col)
{
    int count = 0;           /* counter of living neighbors                    */
    for (int i = row-1; i <= row+1; i++)
        for (int j = col-1; j <= col+1; j++)
            if (map[i][j])       count++;
    if (map[row][col])       count--;
```

```
    return count;
}

void Initialize(bool map[][MAXCOL+2], int mode)
{
    FILE *inf;

    int row, col;                   /* coordinates of a cell    */
    switch (mode)
    {
    case 1: inf = fopen("Virus.dat","r");      break;
    case 2: inf = fopen("Barber_Pole.dat","r");    break;
    case 3: inf = fopen("blinker.dat","r");    break;
    case 4: inf = fopen("blinker2.dat","r");    break;
    case 5: inf = fopen("glider.dat","r");      break;
    case 6: inf = fopen("Glider_Gun.dat","r");    break;
    case 7: inf = fopen("Tumbler.dat","r");    break;
    case 8: inf = fopen("Tumbler2.dat","r");    break;
    case 9: inf = fopen("R_Pentomino.dat","r");    break;
    case 10: inf = fopen("Cheshire_Cat.dat","r");    break;
    case 11: inf = fopen("Harvester.dat","r");    break;
    default:    break;
    }

    while(1)
    {
        fscanf(inf,"%d %d", &row, &col);
        if    (row != 0 || col != 0)        map[row][col] = 1;
        else break;
    }
}

void WriteMap(bool map[][MAXCOL+2])
{
    for (int row = 1; row <= MAXROW; row++)
    {
        for (int col = 1; col <= MAXCOL; col++)
        {
            if (map[row][col])              ezdSetColor(ezdRed);
            else                            ezdSetColor(ezdLightBlue);
```

```
                ezdDrawPoint(VSx(col), VSy(row));
                //ezdDrawRectangle(VSx(col), VSy(row), VSx(col+1), VSy(row+1));
            }
        }
}

void CopyMap(bool map[][MAXCOL+2], bool newmap[][MAXCOL+2])
{
    int row, col;
    for (row = 0; row <= MAXROW + 1; row++)
        for (col = 0; col <= MAXCOL + 1; col++)
            map[row][col] = newmap[row][col];
}

void setWindow(float x1, float y1, float x2, float y2)
{
    WS.x1 = x1; WS.y1 = y1; WS.x2 = x2; WS.y2 = y2;
    openWindow();
    mfx = 10./(WS.x2-WS.x1);    mfy = 8./(WS.y2-WS.y1);
    mfx = mfy = (mfx < mfy ? mfx : mfy);
}

double Rand_0_1(long int *seed)     // 使用高精密之亂數函數
{
    *seed = 16807 * (*seed % 127773) - 2836 * (*seed/127773);
    if (*seed < 0)      *seed += 2147483647;
    return (double) *seed / (double) 2147483647;
}

void DrawLine(POSI p1, POSI p2, shapeColor c)
{
    lineShape Line(VSx(p1.x), VSy(p1.y),VSx(p2.x), VSy(p2.y), c) ;
    Line.draw();
}
```

三、河內塔 Hanoi Tower

河內之塔(Towers of Hanoi)是法國人M.Claus(Lucas)於1883年從泰國帶至法國的，河內為越戰時北越的首都，即現在的胡志明市；1883年法國數學家 Edouard Lucas 曾提及這個

故事，據說創世紀時 Benares 有一座波羅教塔，是由三支鑽石棒（Peg）所支撐，開始時神在第一根棒上放置 64 個由上至下依由小至大排列的金盤（Disc），並命令僧侶將所有的金盤從第一根石棒移至第三根石棒，且搬運過程中遵守大盤子在小盤子之下的原則，若每日僅搬一個盤子，則當盤子全數搬運完畢之時，此塔將毀損，而也就是世界末日來臨之時。

不管這個傳說的可信度有多大，如果考慮一下把 64 片金盤，由一根鑽石棒上移到另一根鑽石棒上，並且始終保持上小下大的順序。這需要多少次移動呢?

3.1 河內塔的搬法規則

設存在 A、B、C 三個柱子〈peg〉以及 n 個大小均不同的套環〈disc〉，起初 n 個套環均由大而小的全部套置於柱子 A 上，今欲將 n 個套環以最少步全部由柱子 A 移到柱子 C，柱子 B 可當作其中間橋樑其運作過程需符合下列規則：

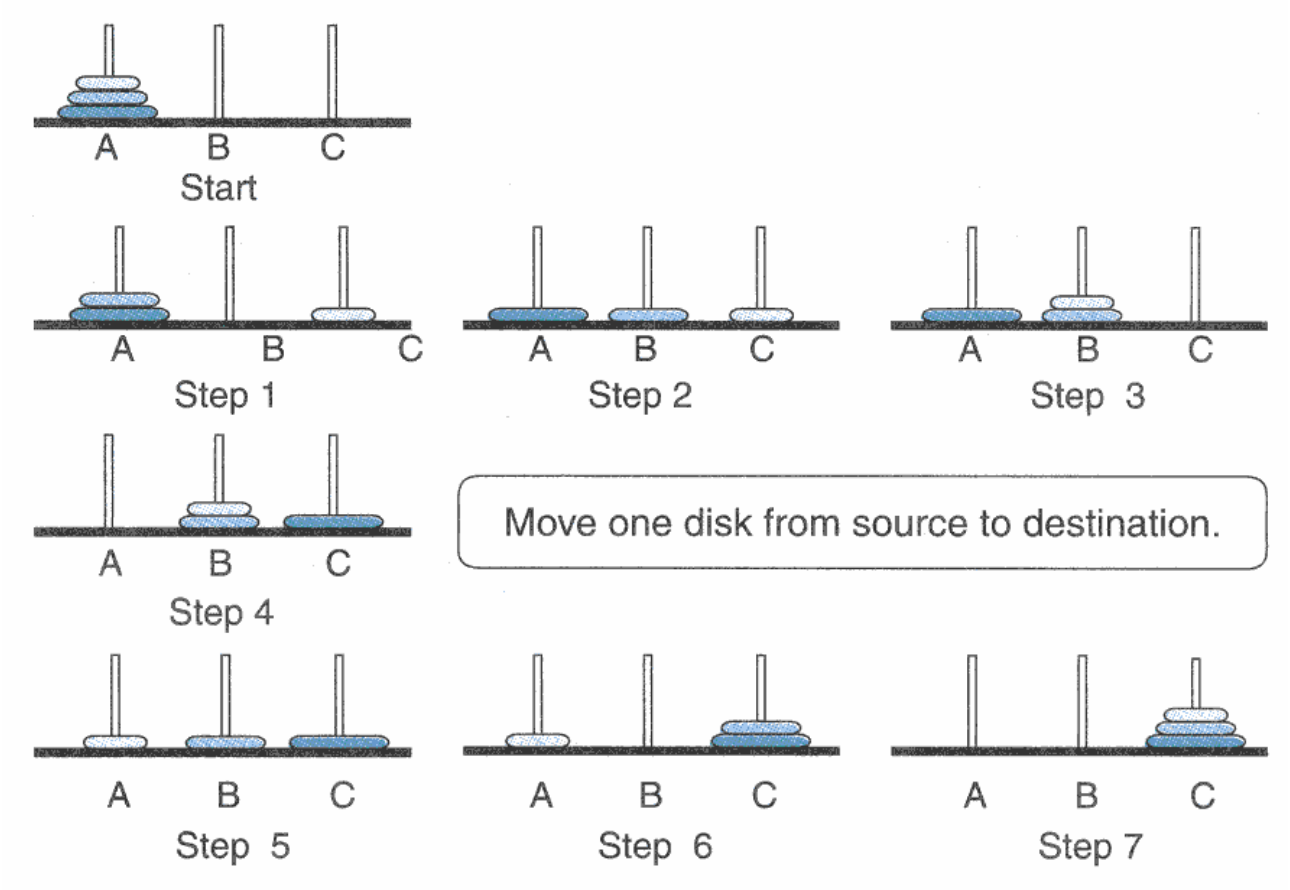
1. 直徑較小的套環永遠置於直徑較大的套環上
2. 套環可恣意地由任何一個柱子移到其他的柱子上
3. 每一次僅能移動一個套環
4. 下圖為套環初始配置圖

為了分辨 n 個十小不同的套環，我們將其由小而大依序編號為 1,2,3,...,n-1,n。接著，我們評估其所須的移動次數及次序，茲以下述 n=1,n=2 及 n=3 來求出其規則性。

1. 當 **n=1**,套環移動次序如下:
   第 1 步**移動套環 1 從柱子 A 到柱子 C。**
   因此總共須移動 2-1=1 步套移動次序為 **1**。
2. 當 n=2 套環移動次序如下:
   第 1 步移動套環 1 從柱子 A 到柱子 B。
   第 2 步**移動套環 2 從柱子 A 到柱子 C。**
   第 3 步移動套環 1 從柱子 B 到柱子 C。
   因此,總共須移動 2-1=3 步，套環移動次序為 1,**2**,1。
3. 當 n=3,套環移動次序如下:
   第 1 步移動套環 1 從柱子 A 到柱子 C。
   第 2 步移動套環 2 從柱子 A 到柱子 B。
   第 3 步移動套環 1 從柱子 C 到柱子 B。
   第 4 步**移動套環 3 從子 A 到柱子 C。**
   第 5 步移動套環 1 從柱子 B 到柱子 A。
   第 6 步移動套環 2 從柱子 B 到柱子 C。

第 7 步移動套環 1 從柱子 A 到柱子 C。

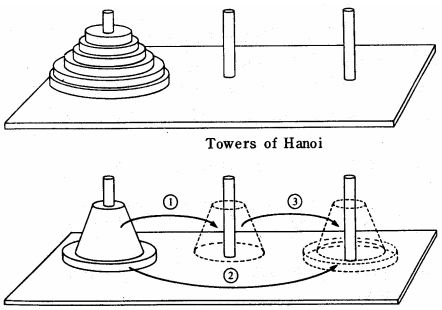
因此，總共須移動 2-1=7 步，套環移動次序為 1,2,1,**3,**1,2,1。



從 n=1 及 n=2 的移動情形，我們可歸納出一個結論即 n=2 是處理 n=1 兩次，總共須移動 2^1-1=3 步，其漢諾塔數列為 121。同理可推知 n=3 是處理 n=2 兩次,總共須移動 2^3-1=7 步其漢諾塔數列為 1213121，同理可推知 n=4 是處理 n=3 兩次，總共須移動 2^4-1=15 步其漢諾塔數列為 121312141213121,餘依此類推.....。

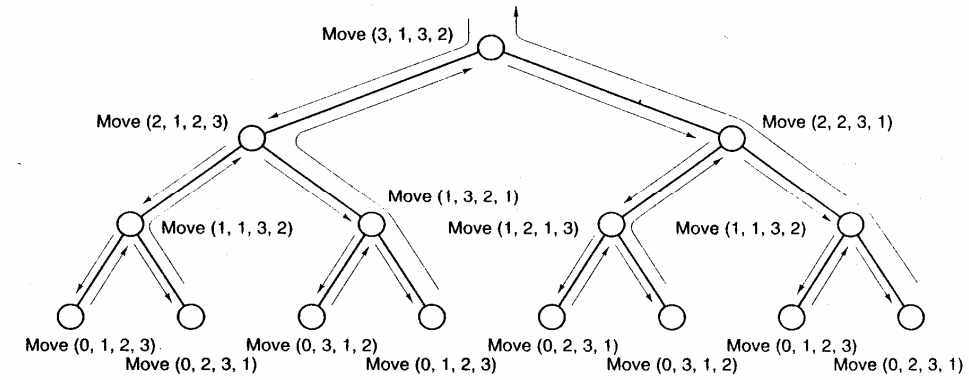
- 接下來請追蹤某一個套環的移動過程，再追蹤另一個套環的移動過程，找出規則。
- 其次，將漢諾塔數列中各個套環第一次出現的移動過程集中起來，找出規則。
- 再來請找環數為奇數與偶數的移動過程的異同
- 最後同學們是否能用分解動作寫出總數為 7 個及 8 個的移動過程呢?

```
void hanoi(unsigned N, char frompeg, char topeg, char
auxpeg)
{
   if (0 == N) return;
      hanoi(N-1, frompeg, auxpeg, topeg);
      move remaining disk from frompeg to topeg;
      hanoi(N-1, auxpeg, topeg, frompeg);
```



}

The recursive tree of a 3-ring problem



Time Complexity: T(n) = 2T(n-1) + 1; T(1) = 1

The problem needs 2^N-1 moves.

```
/* A simple version C program */
// RecHanoi.cpp
#include <stdio.h>
void Towers(unsigned, char, char, char);
void main()
{
      unsigned n;
      do {
            printf("Enter ring number (1 < n < 7): ");
            scanf("%d", &n);
      } while (n < 2 || n > 6);
      Towers(n, 'A', 'C', 'B');
}

void Towers (unsigned n, char frompeg, char topeg, char auxpeg)
{
      if (1 == n)
      {
            printf("\n%s%c%s%c", "move disk 1 from peg ", frompeg, " to peg ",topeg);
            return;
```

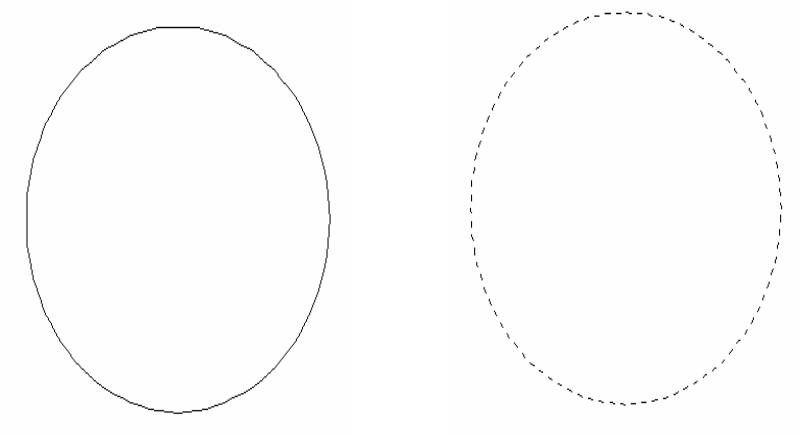
```
    }
    Towers(n-1, frompeg, auxpeg, topeg);
    printf("\n%s%d%s%c%s%c", "move disk ", n, " from peg ", frompeg, " to peg ",topeg);
    Towers(n -1, auxpeg, topeg, frompeg);
}
```

```
#include <stdio.h>
#include "d_draw.h"      // graphics library, openWindow(); viewWindow();   closeWindow();
#include "d_rectsh.h"

struct WorkingSpace { float x1, y1, x2, y2; } WS;

#define VSx(x)     (((x)-WS.x1))
#define VSy(y)     ((WS.y2-(y)))
#define     N   7                                   /*   設定 ring 的最大個數   */

void setWindow(float, float, float, float);
void Towers(unsigned, char, char, char);
void DrawIniTower(int n);
void MoveRing(int i, char s, char t);
float ya, yb, yc;
float width = 0.4;
rectShape rings[N];

void main()
{
    int n;
    do {
        printf("Enter ring number (1 < n < 7): ");
        scanf("%d", &n);
    } while (n < 2 || n > 6);
    setWindow(0,0, 10, 6);
    DrawIniTower(n);
    ya = n*width+1, yb = yc = 1;
    Towers(n, 'A', 'C', 'B');
    viewWindow();   closeWindow();
}

void Towers (unsigned n, char frompeg, char topeg, char auxpeg)
{
    if (1 == n)
```

```
    {
        printf("\n%s%c%s%c", "move ring 1 from peg ", frompeg, " to peg ",topeg);
        MoveRing(n-1, frompeg, topeg); delayWindow(1);
        return;
    }
    Towers(n-1, frompeg, auxpeg, topeg);
    printf("\n%s%d%s%c%s%c", "move ring ", n, " from peg ", frompeg, " to peg ",topeg);
    MoveRing(n-1, frompeg, topeg); delayWindow(1);
    Towers(n -1, auxpeg, topeg, frompeg);
}

void DrawIniTower(int n)
{
    float base_x, base_y, length, width=0.4;
    shapeColor color=white;
    rectShape ground(VSx(0.5), VSy(1), 9, 0.15, black);
    ground.draw();
    for (int i = 0; i < n; i++)
    {
        length = (i+1) * width;
        rings[i].setSides(length, width);
        base_x = 2.5-0.5*rings[i].getLength();
        base_y = (n-i)*width+1;
        rings[i].setColor(++color);
        rings[i].move(VSx(base_x), VSy(base_y));
        rings[i].draw();
    }
    delayWindow(1);
}

void MoveRing(int i, char from, char to)
{
    float baseX, baseY;

    switch(from)
    {
    case 'A':   ya -= width;    break;
    case 'B':   yb -= width;    break;
    case 'C':   yc -= width;    break;
    default:    break;
```

```
        }

        switch(to)
        {
        case 'A':
                baseX = 2.5 - 0.5*rings[i].getLength();
                ya += width;            baseY = ya;             break;
        case 'B':
                baseX = 5 - 0.5*rings[i].getLength();
                yb += width;            baseY = yb;             break;
        case 'C':
                baseX = 7.5 - 0.5*rings[i].getLength();
                yc += width;            baseY = yc;             break;
        default:    break;
        }
        rings[i].erase();
        rings[i].move(VSx(baseX), VSy(baseY));
        rings[i].draw();
}

void setWindow(float x1, float y1, float x2, float y2)
{
        WS.x1 = x1; WS.y1 = y1; WS.x2 = x2; WS.y2 = y2;
        openWindow();
}
```

作業 3: 寫出一個可以畫出如下圓形的程式:



```
#include <stdio.h>
#include <math.h>
#include "d_shape.h"      // include d_draw.h
      // graphics library, ezdraw.h , openWindow(); viewWindow();    closeWindow(), ;
#include "d_linesh.h"            // lineShape class

struct WorkingSpace { float x1, y1, x2, y2; } WS;
struct POSI { double x, y; } ;
#define RD (0.0174533)    // RD = 3.14159/180
#define SIN(x)             (sin((x)*RD))
#define COS(x)              (cos((x)*RD))
#define VSx(x)      (((x)-WS.x1)*mfx)
#define VSy(y)      ((WS.y2-(y))*mfy)

void DrawLine(POSI p1, POSI p2, shapeColor c);
void setWindow(float x1, float y1, float x2, float y2);
void Circle(POSI base, float radius, int n, shapeColor c);
void DashCircle(POSI base, float radius, int n, shapeColor c);
POSI MoveTo(POSI base, float length, float deg);

float mfx, mfy;
void main()
{
      POSI base = {2, 4};
      setWindow(-1, 0, 9, 8);
      Circle(base, 1.5, 300, black);
      base.x = 6;
      DashCircle(base, 1.5, 100, red);
      viewWindow();    closeWindow();
}
```

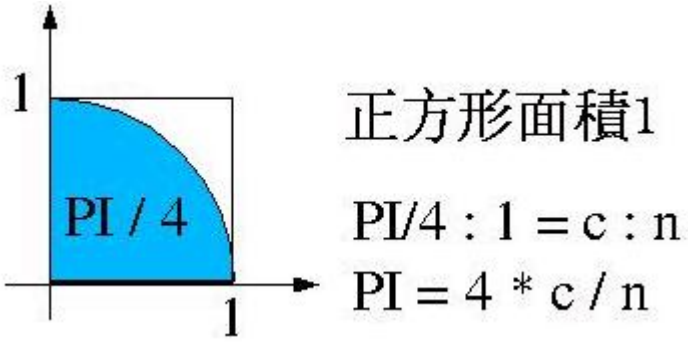
```
/*      畫圓，以 base 為圓心、radius 為半徑，劃 n 段虛線    */
void DashCircle(POSI base, float radius, int n, shapeColor c)
{
    static POSI p1, p2;    float angUnit = 360./n;
    register unsigned i;
    p1.x = base.x + radius;    p1.y = base.y;
    for (i = 1; i <= n; i++)
    {
        p2 = MoveTo(base, radius, i*angUnit);
        if (i & 1) DrawLine(p1, p2, c);
        p1 = p2;
    }
}

void Circle(POSI base, float radius, int n, shapeColor c)
{
    static POSI p1, p2;    float angUnit = 360./n;
    register unsigned i;
    p1.x = base.x + radius;    p1.y = base.y;
    for (i = 1; i <= n; i++)
    {
        p2 = MoveTo(base, radius, i*angUnit);
        DrawLine(p1, p2, c);
        p1 = p2;
    }
}

void setWindow(float x1, float y1, float x2, float y2)
{
    WS.x1 = x1; WS.y1 = y1; WS.x2 = x2; WS.y2 = y2;
    openWindow();
    mfx = 10./(WS.x2-WS.x1);
    mfy = 8./(WS.y2-WS.y1);
    mfx = mfy = (mfx < mfy? mfx: mfy);
}

void DrawLine(const POSI p1, const POSI p2, shapeColor c)
{
    lineShape Line(VSx(p1.x), VSy(p1.y), VSx(p2.x), VSy(p2.y), c);
```

```
    Line.draw();
}

POSI MoveTo(POSI base, float length, float deg)   //以 base 為起點，沿 deg 角度移動 L 距離
{
    POSI vp = {base.x + length*COS(deg), base.y + length*SIN(deg)};
    return vp;
}
```

作業 4: 蒙地卡羅法求解π

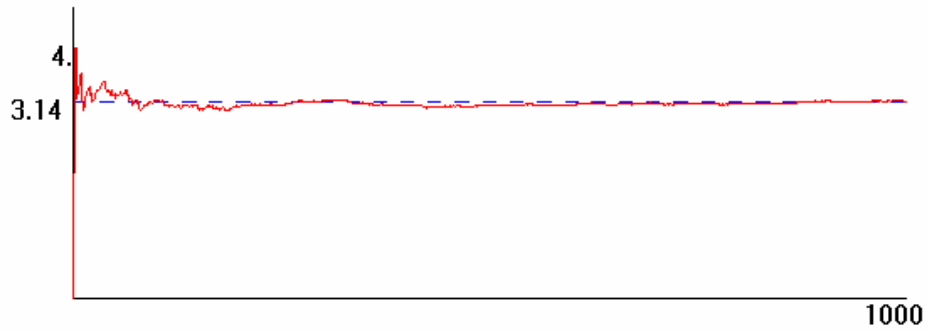
蒙特·卡羅方法（***Monte Carlo method***），也稱**統計模擬方法**，是二十世紀四十年代中期由於科學技術的發展和電子電腦的發明，而被提出的一種以機率統計理論為指導的一類非常重要的數值計算方法。是指使用隨機數（或更常見的偽隨機數）來解決很多計算問題的方法。蒙特·卡羅方法的名字來源於摩納哥的一個城市 Monte Carlo，該城市以賭博業聞名，而蒙特·卡羅方法正是以機率為基礎的方法。與它對應的是確定性演算法。

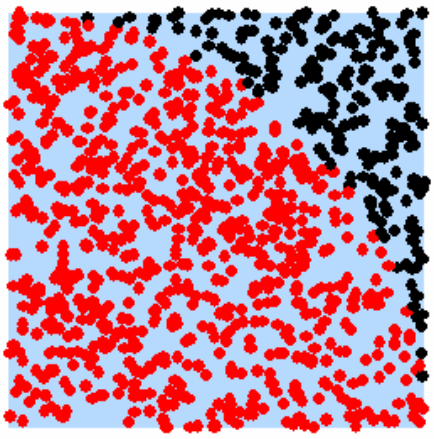
假設有一個圓半徑為 1，所以四分之一圓面積就為 PI，而包括此四分之一圓的正方形面積就為 1，如下圖所示：



如果隨意的在正方形中投射飛標（點）好了，則這些飛標（點）有些會落於四分之一圓內，假設所投射的飛標（點）有 n 點，在圓內的飛標（點）有 c 點，則依比例來算，就會得到上圖中最後的公式。

至於如何判斷所產生的點落於圓內，很簡單，令亂數產生 X 與 Y 兩個數值，如果 $X^2+Y^2<1$ 就是落在圓內。





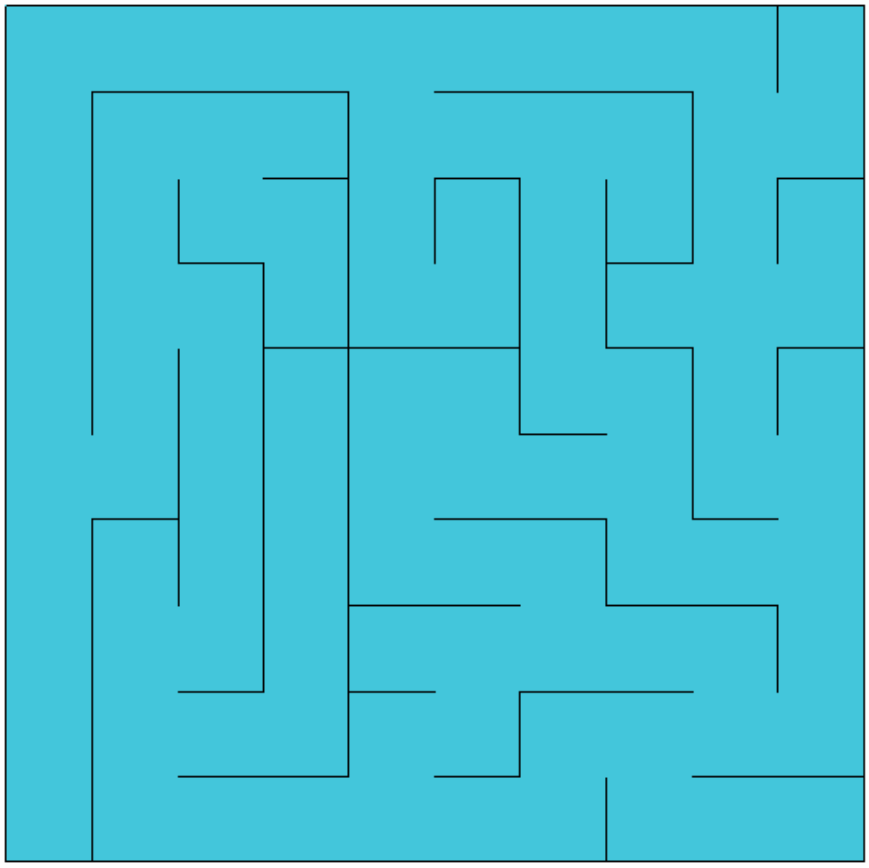
```
// PIbyRand.cpp
#include <time.h>
#include <stdlib.h>
#include <stdio.h>
#include "d_draw.h"       // graphics library, openWindow(); viewWindow();    closeWindow();
#include "d_linesh.h"     // lineShape class
#include "d_textsh.h"
#include "d_rectsh.h"
struct WorkingSpace { float x1, y1, x2, y2; } WS;

struct POSI { double x, y; };

double Rand_0_1(long int *seed);
double PI_Rand();
void      setWindow(float, float, float, float);
void DrawLine(POSI p1, POSI p2, shapeColor c);
void DashLine(POSI star, POSI end, int n, shapeColor c) ;
void DrawNum(float x, float y, float NUM, unsigned n, shapeColor c);
#define VSx(x)      (((x)-WS.x1)*mfx)
```

```
#define VSy(y)      ((WS.y2-(y))*mfy)
#define num 1000
POSI P[num];
float mfx, mfy;

void main()
{
      setWindow(-0.2,-0.2, 2.2, 2.2);
      printf("n = %d pi=%f\n", num, PI_Rand());
      viewWindow();    closeWindow();
}

double PI_Rand()
{
      unsigned long hits = 0;    static double x, y;
      float pi ;
      static char CHAR[80];
      long int seed;
      time(&seed);
      srand((unsigned int)(time(NULL)%10000));
      POSI p0={0., 3.14*0.15+1.3}, p1={2, 3.14*0.15+1.3};       DashLine(p0, p1, 30, blue);
      p0.x = 0, p0.y = 1.3;    p1.x = 2, p1.y = 1.3;   DrawLine(p0, p1, black);
      p0.x = 0, p0.y = 1.3;    p1.x = 0, p1.y = 2;     DrawLine(p0, p1, black);
      DrawNum(-0.15, 3.14*0.15+1.32, 3.14, 4, black);
      DrawNum(-0.05, 4*0.15+1.32, 4, 2, black);
      DrawNum(2-0.1, 1.3, num, 4, black);
      P[0].x = 0,    P[0].y = 1.3;
      rectShape square(VSx(0), VSy(1), 1.*mfy, 1.*mfx , lightblue);    square.draw();

      for (unsigned long i = 1; i <= num; i++)
      {
            x = Rand_0_1(&seed);
            y = Rand_0_1(&seed);
            ezdSetColor(ezdBlack);
            if (x * x + y * y < 1.0)
            {
                  ++ hits;
                  ezdSetColor(ezdRed);
            }
            ezdDrawPoint(VSx(x), VSy(y));
```

```
            P[i].x = P[i-1].x + 2./num;
            pi = 4.0 * hits / i;
            P[i].y = pi * 0.15 +1.3;
            DrawLine(P[i-1], P[i], red);
            delayWindow(30./num);
        }
        return pi;
}

double Rand_0_1(long int *seed)     // 使用高精密之亂數函數
{
        *seed = 16807 * (*seed % 127773) - 2836 * (*seed/127773);
        if (*seed < 0)      *seed += 2147483647;
        return (double) *seed / (double) 2147483647;
}

void setWindow(float x1, float y1, float x2, float y2)
{
        WS.x1 = x1; WS.y1 = y1; WS.x2 = x2; WS.y2 = y2;
        openWindow();
        mfx = 10./(WS.x2-WS.x1);    mfy = 8./(WS.y2-WS.y1);
        mfx = mfy = (mfx < mfy ? mfx : mfy);
}

void DrawLine(POSI p1, POSI p2, shapeColor c)
{
        lineShape Line(VSx(p1.x), VSy(p1.y),VSx(p2.x), VSy(p2.y), c) ;
        Line.draw();
}

void DashLine(POSI star, POSI end, int n, shapeColor c)
{
        int i = 0;
        static lineShape Line; Line.setColor(c);
        star.x = VSx(star.x);
        star.y = VSy(star.y);
        end.x = VSx(end.x);
        end.y = VSy(end.y);

        float dx = (end.x-star.x)/(2*n + 1), dy = (end.y - star.y)/(2*n + 1);
```

```
        Line.move(star.x, star.y);
        do   {
             star.x += dx;    star.y += dy;
             Line.setEndPoint(star.x, star.y);
             Line.draw();
             star.x += dx;    star.y += dy;
             Line.move(star.x, star.y);
        } while(++i <= n);
}

void DrawNum(float x, float y, float NUM, unsigned n, shapeColor c)
{   // NUM is the number sting of n digits to be output
        static char CHAR[80];
        textShape text(VSx(x), VSy(y), gcvt(NUM,n, CHAR), c);
        text.draw();
}
```

作業 6:寫出一個可以畫出如下迷宮的程式:



```
/*************************************************************************/
/*      This program was written by Jing Lee on 2007/8/16. This program draws a maze.
/*      The maze data is stored in data files: Maze1.dat, Maze2.dat, Maze3.dat, Maze4.dat.
/*      There are seven strategies can be chosen.
*/
/*************************************************************************/

#include <stdio.h>
#include "d_draw.h"
#include "d_linesh.h"
#include "d_textsh.h"

#define DataFile "Maze1.dat"          // Give the name of Input File

/*************************************************************************/
// Each cell is defined by a 4-bit binary code
// If the cell code is 0000, then the cell is open.
// If the first (leftmost) bit is 1, then there is a wall in the north direction
// If the second bit is 1, then there is a wall in the west direction
// If the third bit is 1, then there is a wall in the south direction
// If the rightmost bit is 1, then there is a wall in the east direction
/*************************************************************************/

#define VSx(x)              (((x)-WS.x1)*mfx)
#define VSy(y)              ((WS.y2-(y))*mfy)
#define ODD(m)               ((m) & 1)

/*************************************************************************/
// The following functions are used to draw the graph
typedef struct Point2Struct { double x, y;} Point2;
typedef struct Position { unsigned r, c;} Location;
struct WorkingSpace { float x1, y1, x2, y2; } WS;

void setWindow(float x1, float y1, float x2, float y2);
void DrawLine(Point2 p1, Point2 p2);
void DrawNum(Location base, int NUM, unsigned n, shapeColor c);
void DrawCell(Location base, unsigned value);
void DrawMaze(unsigned**, unsigned, bool);
unsigned** ReadMazeData(unsigned&);
```

```
lineShape Line(0,0,0,0,black);
float mfx, mfy;

void main()
{
	unsigned SIZE;
	unsigned **MazeData = ReadMazeData(SIZE);
	setWindow(0, 0, SIZE, SIZE);
	DrawMaze(MazeData, SIZE, 0);
		// 0: do'nt output code number; 1: output code number of cell
	viewWindow();	closeWindow();
}

unsigned** ReadMazeData(unsigned& SIZE )
{
	FILE *in = fopen(DataFile, "r");
	fscanf(in, "%d ", &SIZE);	SIZE += 2;
	// For simplying the problem, we add sentinels at the 4 boundaries of the maze
	// So the matrix size musts 2 rows and 2 columns larger than the practical maze

	unsigned **MazeData = (unsigned **)new unsigned*[SIZE];
	for (unsigned i = 0; i < SIZE; i++) MazeData[i] = (unsigned *) new unsigned[SIZE];
	unsigned r, c;
	for (r = SIZE-2; r >= 1; r--)
		for (c = 1; c < SIZE-1; c++)	fscanf(in,"%d ", &MazeData[r][c]);

	//	Add sentinals at the four boundaries of the maze
	for (c = 0; c < SIZE; c++)
		MazeData[0][c] = MazeData[SIZE-1][c] = 15;	// Add sentinels
	for (r = 0; r < SIZE; r++)
		MazeData[r][0] = MazeData[r][SIZE-1] = 15;	// Add sentinels
	return MazeData;
}

void setWindow(float x1, float y1, float x2, float y2)
{
	WS.x1 = x1; WS.y1 = y1; WS.x2 = x2; WS.y2 = y2;
	openWindow();
	mfx = 10./(WS.x2-WS.x1);	mfy = 8./(WS.y2-WS.y1);
	mfx = mfy = (mfx < mfy ? mfx : mfy);
```

```
}

void DrawMaze(unsigned **A, unsigned SIZE, bool flag)
{
      Location p;
      ezdSetColor(ezdTurquoise);
      ezdDrawRectangle(VSx(1), VSy(1), VSx(SIZE-1), VSy(SIZE-1));
      for (p.r = 1; p.r < SIZE-1; p.r++)
            for (p.c = 1; p.c < SIZE-1; p.c++)
            {
                  DrawCell(p, A[p.r][p.c]);
                  if (flag)          DrawNum(p, A[p.r][p.c], 3, darkgray);
            }
}

void DrawCell(Location base, unsigned value)
// base is the left-low point of the cell
{
      Point2 LL, RL, RU, LU;    // LL: left-low point, RL: right-low, RU: right-up, LU: left-up
      LL.x = RL.x =    LU.x = base.c;
      LL.y = RL.y =    LU.y = base.r;
      RL.x = LL.x + 1;
      RU.x = RL.x;      RU.y = RL.y + 1;
      LU.y = LL.y + 1;
      Line.setColor(black);
      if (ODD(value))             DrawLine(RL, RU);      // East wall
      if (ODD(value >> 1))    DrawLine(LL, RL);      // South wall
      if (ODD(value >> 2))    DrawLine(LL, LU);      // West wall
      if (ODD(value >> 3))    DrawLine(RU, LU);     // North wall
}

void DrawLine(Point2 p1, Point2 p2)
{
      Line.move(VSx(p1.x), VSy(p1.y));
      Line.setEndPoint(VSx(p2.x), VSy(p2.y));
      Line.draw();
}

void DrawNum(Location base, int NUM, unsigned n, shapeColor c)
{    // NUM is the number sting of n digits to be output
```

```
    static char CHAR[3];
    textShape text(VSx(base.c+0.2), VSy(base.r+0.7), gcvt(NUM, n, CHAR),c);
    text.draw();
}
```

神秘的迷宮

迷宮最早出現在古希臘神話中。據說，半人半神的英雄西修斯 (Theseus) 在克裏特的迷宮中勇敢地殺死半人半牛的怪物，並循著繩索 (Ariadne 抓住另一頭) 逃出迷宮。 希臘史學家希羅多德曾探訪過那裏。他描述說，整個迷宮由 12 座帶頂院落構成，所有院落都由通道連接，形成 3000 個獨立的"室"。據說，建造這座迷宮用的人力和財力"超過了希臘所有的建築"。後來的參觀者也說，一旦進入迷宮，如果沒嚮導，根本無望走出。若不是知情人洩露了地圖，盜墓者可能永遠也無法探明克裏特迷宮。

歷史上 , 人們認為迷宮具有魔力。中國，有三國時諸葛亮預先布下八卦陣，在劉備兵敗時，東吳陸遜被困於八卦陣的故事。杜甫為此有詩曰：「功蓋三分國，名成八陣圖，江流石不轉，遺恨失吞吳。」 留給後人無盡的想像。

金庸小說射雕英雄傳中，精通陰陽五行的黃藥師的桃花島就是個大迷宮。小龍女與楊過幼時居住的古墓，也是個複雜的迷宮。

阿美族傳說中，很早以前曾遭一群身材高大、法術高強自稱「Alikakay」(阿里卡該)的異類入侵。部落裡時常發生一些離奇失蹤、死亡的事件。所以族裡就聚集了村裡最優秀的青年組成了阿美族的勇士軍隊來討伐「阿里卡該」，這些勇士都要受到嚴格的訓練，包括跑步、跳遠、射箭、刀術、投石、原野戰等技能，來打敗阿里卡該並保護村莊的安全。現在，光復鄉的原野迷宮邀請大家來成為阿美族的勇士，接受迷宮園區的山訓設施的訓練，通過迷宮考驗，一起來打敗「阿里卡該」，成為光榮的勇士吧！（迷宮的人形圖騰即為紀念當時犧牲奉獻的勇士們)

英國也有作家豪斯各特，他的小說《皇家獵宮》中，英皇為他的情人羅莎蒙德建造一座神奇迷宮，但最後皇后找到了羅的藏身之所，逼其自殺。

後來，迷宮成為遊戲。歐洲的宮廷和公園裏就有許多迷宮，如今，很多歐洲人自己家中也出現了迷宮。 炎炎夏日，正是向日葵開花的季節。在德國的施托克，數十萬株向日葵排成龐大的迷宮。迷宮主人備有觀光氣球，坐上觀光熱氣球，從高空俯瞰，條條路徑穿梭在這個占地 2.2 萬平方米 的向日葵迷宮間，組成一幅幅“金色花海”的幾何圖形。回到地面，一頭鑽進迷宮，一路欣賞著搖曳多姿的花朵，享受著四周的新鮮空氣，雖迷路多時，卻不急著尋找出口。

除了向日葵迷宮，歐洲還有很多壯觀的麥田迷宮。然而植物迷宮雖充滿野外樂趣，但受季節限制。平日更流行的是傳統的花園迷宮，包括草坪迷宮、樹籬迷宮等。近年來，主題迷宮受到越來越多歐洲人的青睞，嗅覺迷宮、聲音迷宮.....形形色色，數不勝數。倫敦有一個鏡子迷宮，是由鏡子、閃爍的燈光和音響構成的，有 1000 平方米 。初次行走，一般要用數小時才走出。

德國人費舍爾是世界上惟一的全職迷宮製造商。 52 歲的費舍爾和他的 12 名員工負責為 25 個國家製造迷宮。他們造的 400 多個迷宮遍佈全球，包括美國的城堡迷宮、英國的龍迷宮，還有將於 2008 年奧運會前在中國開放的茉莉花迷宮等。他說：“迷宮設計不僅

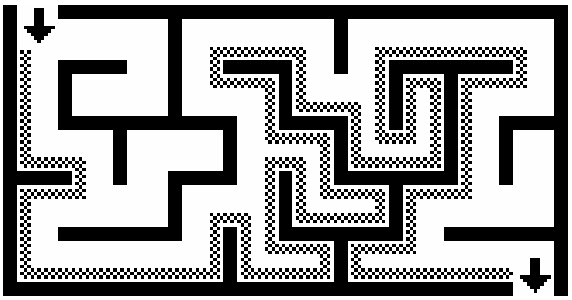
是線條和圖案的組合，要有娛樂性、裝飾性，還要考慮那些穿越迷宮者的感受，那樣才趣味無窮。”

尋找出路

數學家歐拉是首位使用平面分析方法來尋找迷路園出路的人，此研究被稱為拓撲學。以下的演算法是為不知道迷路園設計的挑戰者在身處迷路園時尋路用。

使用右手定律的遍歷

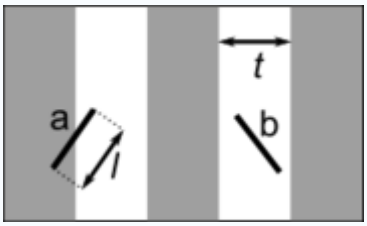
- 右手定律，或名為左手定律是最著名的迷路園演算法。如果該迷路園是由簡單的連接組成，換句話說，即是所有牆皆連接起來，這樣只要將其中一隻手依著其中一堵牆，然後前進，則肯定不會迷路，而且可尋到出口，否則將會有機會回到原點。但如果該迷路園並非簡單連接，則這個方法不可助您找到迷路園的關節位。右手定律亦可以解決三維或更高次元空間的迷路園。



參考資料:

1. W. Ford and W. Topp, Data Structures with C++ Using STL, Prentice Hall, 2002

布豐投針問題



18 世紀，布豐提出以下問題：設我們有一個以平行且等距木紋鋪成的地板（如右圖），現在隨意拋一支長度比木紋之間距離小的針，求針和其中一條木紋相交的機率。這就是**布豐投針問題**。使用積分幾何能找到此題的解，並得出一個求 π 的蒙特·卡羅方法。

設針的長度是ℓ，平行線之間的距離為 *t*，*x* 為針的中心和最近的平行線的距離，θ 為針和線之間的銳角。

$x \in [0, t/2]$ 的機率密度函數為$\frac{2}{t}\,dx$。

$\theta \in [0, \pi/2]$ 的機率密度函數為$\frac{2}{\pi}\,d\theta$。

x,θ 兩個隨機變數互相獨立，因此兩者結合的機率密度函數只是兩者的積：

$$\frac{4}{t\pi}\,dx\,d\theta$$。

當 $x \le \frac{\ell}{2}\sin\theta$，針和線相交。

求上式的積分，得針與線相交的機率：

$$\int_0^{\frac{\pi}{2}} \int_0^{(\ell/2)\sin\theta} \frac{4}{t\pi}\, dx\, d\theta = \frac{2\ell}{t\pi}.$$

拋 n 支針，其中有 h 支針與線相交的機率是：

$$\frac{h}{n} = \frac{2\ell}{t\pi},$$

由此可求得 π：

$$\pi = \frac{2\ell n}{th}$$

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#ifndef RAND_MAX
#define RAND_MAX 2147483648
#endif

void main()
{
	const int n = 5000;
	const int iprt = 1000;
	float **R, w, v, prob, pi2;
	int i, j, m = 0;

	R = calloc((n+1), sizeof(float *));
	for (i = 1; i <= n; i++)	R[i] = calloc((3), sizeof(float));
	pi2 = 2 * atan(1.0);
	for (i = 1; i <= n; i++)
		for (j = 1; j <= 2; j++)	R[i][j] = (float) rand() / (float) RAND_MAX;
	for (i = 1; i <= n; i++)
	{
```

```
            w = R[i][1];
            v = pi2 * R[i][2];
            if (w <= sin(v))          m++;
            if (!(i % iprt))
            {
                  prob = (float) m / (float) i;
                  printf("i = %d, prob = %f, 2/pi = %f\n", i, prob, 1/pi2);
            }
      }
}
```