# 《计算机图形学与数字地图》 上机实验报告（2019级）

**姓名** 武成龙

**班级** 地信19-1班

**学号** 07192329

**环境与测绘学院**

实验一：金刚石图案的绘制

1. 实验目的

对CDC类有初步了解。

1. 实验内容

1、利用 CDC 类提供的功能实现金刚石图案的绘制

2、利用 CDC 类的文本绘制功能添加一个“金刚石”的文本

1. 实验思路

根据圆的参数方程计算每个点的x,y坐标值。建立两个数组，分别存储每个点的x,y坐标值，而后利用两层for循环多次调用moveto,lineto函数，依次画出每个线条，需要注意每个点只需要与它后面的点连线。所以最后一个点不需要再连线。

1. 关键代码

对OnDraw函数添加代码：

void C地信191班071923291View::OnDraw(CDC\* pDC)

{

C地信191班071923291Doc\* pDoc = GetDocument();

ASSERT\_VALID(pDoc);

if (!pDoc)

return;

//定义圆周率，圆半径，输出文字

double pi = 3.1415926;

int R = 300;

//创建画笔，定义画笔，选择画笔

CPen my\_pen, \* p\_oldpen;

my\_pen.CreatePen(PS\_SOLID, 1, RGB(230, 0, 0));

p\_oldpen = pDC->SelectObject(&my\_pen);

//画直径为300的圆

pDC->Ellipse(0, 0, 600, 600);

//存放横坐标，纵坐标

int x\_co[30], y\_co[30];

for (int i = 0; i < 30; i++)

{

x\_co[i] = R + R \* cos(2 \* pi / 30 \* i);

}

for (int i = 0; i < 30; i++)

{

y\_co[i] = R + R \* sin(2 \* pi / 30 \* i);

}

//连线。注意i1到29就可以停止，最后一个点不需要再连线

for (int i1 = 0; i1 < 29; i1++)

{

for (int i2 = i1; i2 < 29; i2++)

{

pDC->MoveTo(x\_co[i1], y\_co[i1]);

pDC->LineTo(x\_co[i2 + 1], y\_co[i2 + 1]);

}

}

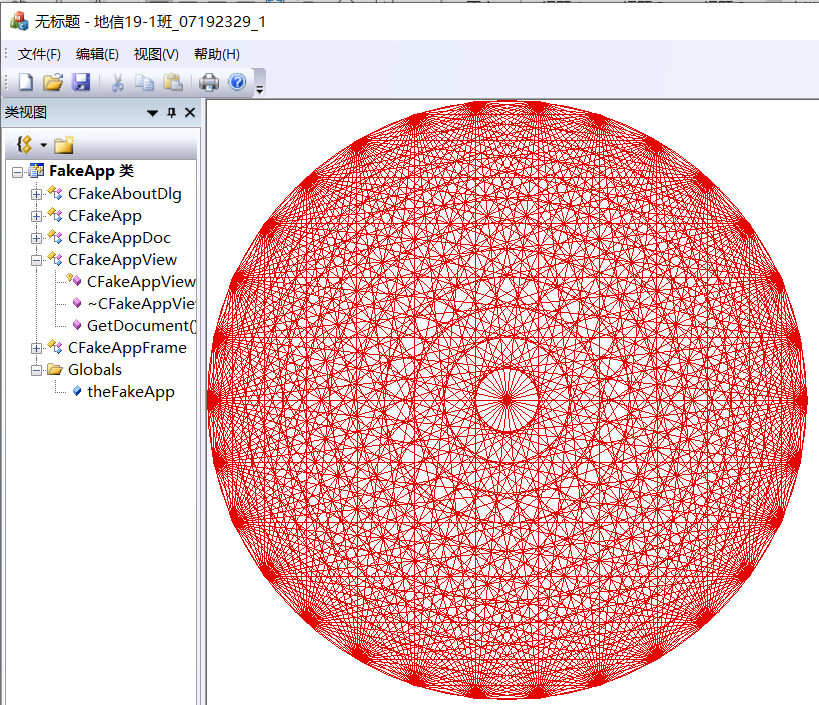
//输出文字

pDC->TextOut(270, 620,L"金刚石");

//删除画笔

my\_pen.DeleteObject();

}

1. 实验结果  
   
2. 实验体会

MFC工程可以方便的进行点，线和各种矢量图形的绘制

实验二：地图数据库与地图绘制

1. 实验目的

了解地图数据库基本原理，掌握地图绘制编程方法

1. 实验内容

1）对于给定的地图数据库，将地图数据库中的线要素及点要素读取出来， 并绘制相应地图至屏幕区域上，如下图所示。

2）将点的名称以注记形式显示在地图上。

1. 实验思路

定义两个类，分别是边境点类和城市点类，把这两个个类的指针作为CDC的成员变量，绘图时直接调用这些点的坐标信息即可。定义一个read函数，用于从文件中读取边境点数据和城市点数据，然后添加鼠标动作，选择菜单栏中的文件-打开菜单绘图。

1. 关键代码

1、在 C地信191班071923292View.h 文件的类文件定义之前添加以下代码

#include <string> //C++字符串类

using namespace std;//命名空间，string类在std命名空间下

struct MyPoint

{

int x;

int y;

int rank;

string name;

int id;

};

2、在 C地信191班071923292View 类中增加新的数据成员 pts

public:

MyPoint\* pChinaPts;//存储China数据

MyPoint\* pCityPts;//存储City数据

int chinaPtsCount;//China的点数

int cityPtsCount;//City 的点数

3、构造函数中初始化

C地信191班071923292View::C地信191班071923292View() noexcept

{

pChinaPts = NULL;

chinaPtsCount = 0;

pCityPts = NULL;

cityPtsCount = 0;

}

4、在 C地信191班071923292View.cpp 文件中包含头文件   
#include <fstream> //C++文件流类，用于文件读取  
5、在CXXView类中自定义两个函数，分别用于China和City文件的读取，并将结果保存至pChinaPts及pCityPts数组中

int C地信191班071923292View::readChinaFromFile(void) {

ifstream ifile("C:\\Users\\10599\\Desktop\\计算机图形学实验\\实验2\\China.txt"); //将文件绑定在一个文件流上

ifile >> chinaPtsCount;//从文件流输出一个整型数据到chinaPtsCount变量

if (pChinaPts) delete[] pChinaPts;//清空指针

pChinaPts = new MyPoint[chinaPtsCount];//动态分配内存

string temp;

ifile >> temp >> temp >> temp >> temp >> temp;//读入第2行，均为字符串,表示各列含义

int id;

for (int i = 0; i < chinaPtsCount; i++)

{

ifile >> pChinaPts[i].id >> temp >> pChinaPts[i].x >> temp >> pChinaPts[i].y;

//读入第i行数据，并保存在pChinaPts数组中

}

return 0;

}

int C地信191班071923292View::readCityFromFile(void) {

ifstream ifile("C:\\Users\\10599\\Desktop\\计算机图形学实验\\实验2\\city.txt"); //将文件绑定在一个文件流上

ifile >> cityPtsCount;//从文件流输出一个整型数据到cityPtsCount变量

if (pCityPts) delete[] pCityPts;

pCityPts = new MyPoint[cityPtsCount];//动态分配内存

string temp;

ifile >> temp >> temp >> temp >> temp >> temp >> temp >> temp >> temp >> temp;

//读入第2行，均为字符串,表示各列含义

int id;

for (int i = 0; i < cityPtsCount; i++)

{

ifile >> pCityPts[i].id >> temp >>

pCityPts[i].name >> temp >>

pCityPts[i].rank >> temp >>

pCityPts[i].x >> temp >>

pCityPts[i].y;//读入第i行数据，并保存在pCityPts数组中

}

return 0;

}  
6、为“文件-打开”菜单添加消息处理函数OnOpenFile（方法见第2章PPT），并 在该函数中做以下操作。  
void C地信191班071923292View::OnFileOpen()

{

readChinaFromFile();

readCityFromFile();

Invalidate();//宣告图形无效

}

7、地图数据绘制，在 OnDraw 函数增加以下代码  
for (int i = 0; i < chinaPtsCount - 1; i++)//绘制china

{

pDC->MoveTo(pChinaPts[i].x, 600 - pChinaPts[i].y);

pDC->LineTo(pChinaPts[i + 1].x, 600 - pChinaPts[i + 1].y);

}

for (int i = 0; i < cityPtsCount; i++)//绘制city

{

pDC->Ellipse(pCityPts[i].x - 2, 600 - pCityPts[i].y - 2, pCityPts[i].x + 2, 600 - pCityPts[i].y + 2);

////////////////////////////////////////////////////////////////////

//////字符串类型转换，将string转为CString

////////////////////////////////////////////////////////////////////

#ifdef \_UNICODE //如果是unicode工程

USES\_CONVERSION; CString tempStr(pCityPts[i].name.c\_str());

#else //如果是多字节工程

CString ans;

tempStr.Format("%s", pCityPts[i].name.c\_str());

#endif // \_UNICODE

////////////////////////////////////////////////////////////////////

pDC->TextOut(pCityPts[i].x + 3, 600 - pCityPts[i].y + 3, tempStr); //输出注记

}

1. 实验结果



1. 实验体会

地图数据库的基本原理就是从数据库中读取相关信息，而后调用绘图语句进行地图绘制。定义一个单独的read函数用于读取文件中的点信息，而后再利用OnDraw绘图，充分体现的面向对象的封装性。

实验三：中点画线算法直线绘制

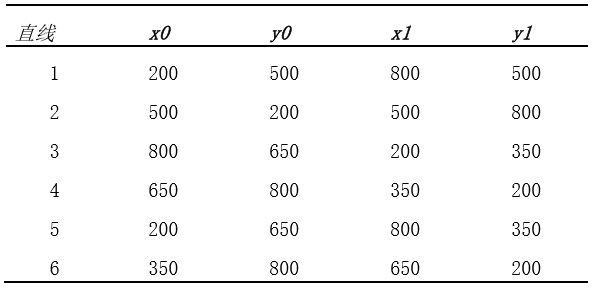
1. 实验目的

1）了解中点画线算法的基本原理；

2）学会使用 VC++实现直线生成算法编程.

1. 实验内容

对于任意输入的两点（斜率任意、起始点顺序任意），利用中点画线算法实 现的直线生成与绘制，要求能正确绘制以下 6 条直线。



1. 实验思路

直线的方向可以分为八个，根据k与1的关系可分为1/4/5/8和2/3/6/7/两组，对于前者x变化更快，应每次变化x，确定y是保持不变还是变化一，后者则是应每次变化y，确定x是保持不变还是变化一；根据起点与终点的关系，可分为4/5/6/7和1/2/3/8两组，第一组交换起点终点后，及和第二组等价。分析过这些后，我们发现可以把八个方向归化为1/8/2/3四个方向，而1/8之间，2/3之间的差别仅在一个负号而已，实质就只需要就1,2两个方向进行分析。

1. 关键代码  
   在头文件中地信19-1\_07192329\_3View.h添加公有成员函数声明：

virtual bool midpoint\_line(CDC\*& pDC, CPoint start\_p, CPoint end\_p, int color);

在源文件地信19-1\_07192329\_3View.cpp中添加相应函数定义：

bool C地信191071923293View::midpoint\_line(CDC \*&pDC,CPoint start\_p, CPoint end\_p, int color)

{

int a, b, d1, d2, d, x, y;

//初始化

int k\_flag = 0;//判断斜率大小，绝对值大于1则值为1

int s\_flag = 1;//判断斜率正负，大于0为1，小于0为-1

//保证从小到大画线

if (start\_p.x>end\_p.x) {

CPoint pt = start\_p;

start\_p = end\_p;

end\_p = pt;

}

//判断斜率与1关系

if (abs(end\_p.y-start\_p.y) > abs(end\_p.x-start\_p.x)) {

k\_flag = 1;

}

//判断斜率与0关系

if (start\_p.y > end\_p.y) {

s\_flag = -1;

}

//保证a是起点-终点

if (s\_flag == -1)

end\_p.y = start\_p.y + (start\_p.y-end\_p.y);

a = start\_p.y - end\_p.y;

b = end\_p.x - start\_p.x;

d = 2 \* a + b;

d1 = 2 \* a;

d2 = 2 \* (a + b);

x = start\_p.x;y = start\_p.y;

//开始画线

pDC->SetPixel(x, y, color);

//与y轴平行的

if (start\_p.x == end\_p.x) {

for (int i = 0; i < abs(start\_p.y - end\_p.y); i++)

{

pDC->SetPixel(x, y + 1, color);

y++;

}

return true;

}

//与x轴平行的

if (start\_p.y == end\_p.y) {

for (int i = 0; i < abs(start\_p.x - end\_p.x); i++)

{

pDC->SetPixel(x + 1, y, color);

x++;

}

return true;

}

//一般情况

if (k\_flag == 0) {//斜率小于1

for (int i = 0; i < (end\_p.x-start\_p.x); i++)

{

if (d < 0) {

x += 1;

y += s\_flag;

d += d2;

pDC->SetPixel(x, y, color);

}

else {

x +=1;

d += d1;

pDC->SetPixel(x, y, color);

}

}

return true;

}

else {//斜率大于1

if (k\_flag == 1) {

d1 = 2 \* b;

d = 2 \* b + a;

}

for (int i = 0; i < abs(end\_p.y - start\_p.y); i++)

{

if (d < 0) {

//x += 1;

y += s\_flag;

d += d1;

pDC->SetPixel(x, y, color);

}

else {

x += 1;

y += s\_flag;

d += d2;

pDC->SetPixel(x, y, color);

}

}

return true;

}

}  
同时，OnDraw函数修改为：

void C地信191071923293View::OnDraw(CDC\* pDC)

{

C地信191071923293Doc\* pDoc = GetDocument();

ASSERT\_VALID(pDoc);

if (!pDoc)

return;

CPoint p1(200, 500),p2(800,500),p3(500,200),p4(500,800),

p5(800, 650),p6(200, 350),p7(650, 800),p8(350, 200),

p9(200, 650),p10(800, 350),p11(350, 800),p12(650, 200);

midpoint\_line(pDC, p1, p2, (0, 0, 0));

midpoint\_line(pDC, p3, p4, (0, 0, 0));

midpoint\_line(pDC, p5, p6, (0, 0, 0));

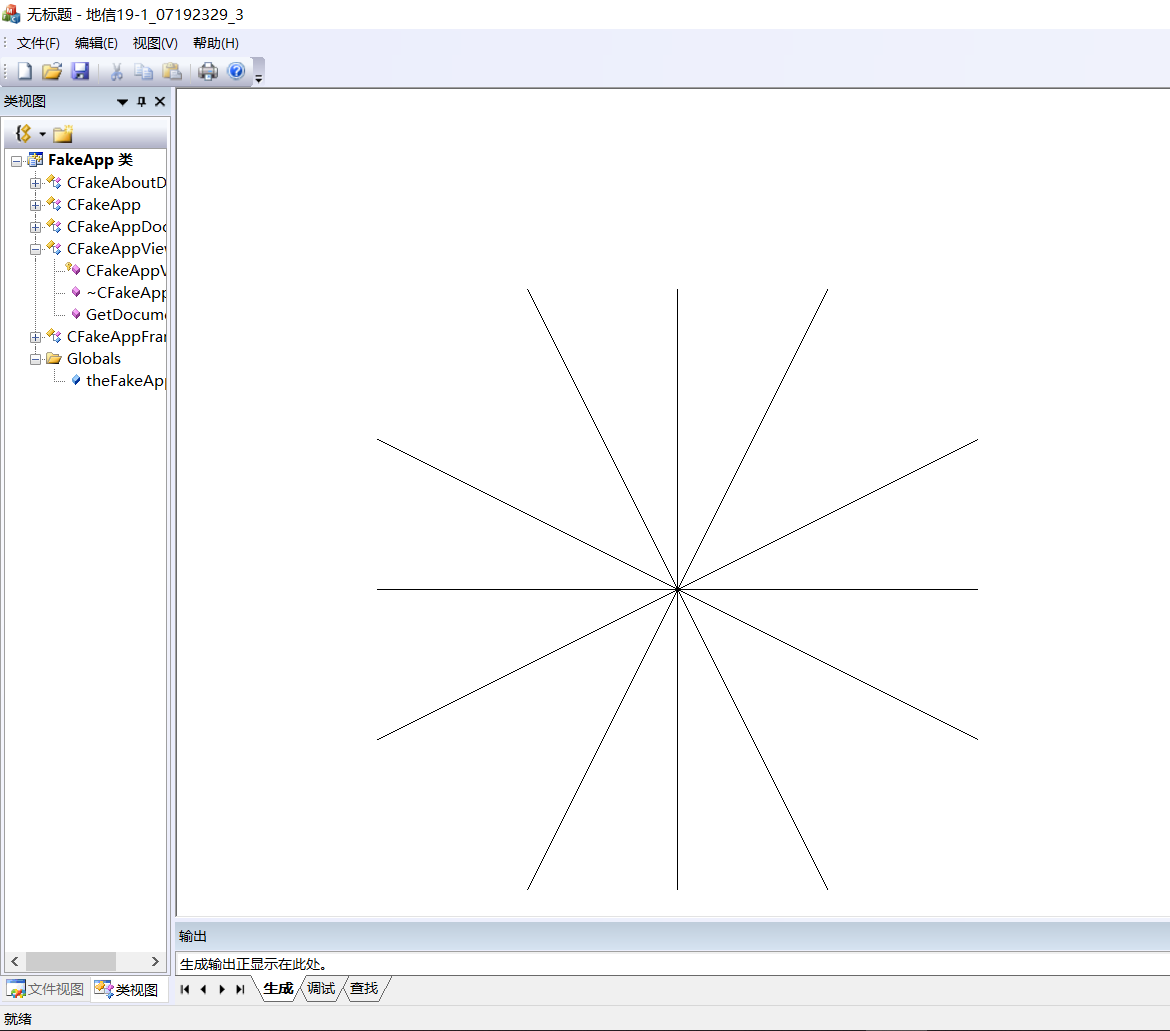
midpoint\_line(pDC, p7, p8, (0, 0, 0));

midpoint\_line(pDC, p9, p10, (0, 0, 0));

midpoint\_line(pDC, p11, p12, (0, 0, 0));

}

1. 实验结果



1. 实验体会

在开始实践之前，先对八个方向进行分析，找到互通之处，对于可以归于一类处理的方向尽量归于一类，可以大大减少分类讨论的工作量。

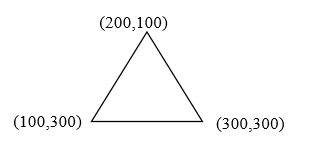
实验四：二维图形变换的编程实现

1. 实验目的

1）了解二维图形变换的基本原理；

2）学会使用 VC++实现图形变换

1. 实验内容
2. 利用实验三的直线绘制功能，绘制下列坐标的三角形。



1. 用程序实现上述三角形关于点（300,300）做顺时针 90°的旋转变换，并 绘制变换后的图形。
2. 实验思路

先定义矩阵乘法函数。而后将三角形平移到原点，而后旋转，再平移回去，就可以得到变换矩阵。将三角形坐标矩阵扩展为三维，与变化矩阵相乘即可得到变化后的点坐标，而后绘制新的三角形

1. 关键代码

在实验三基础上添加如下代码：

头文件地信19-1班\_07192329\_4View.h中添加函数声明

virtual bool midpoint\_line(CDC\*& pDC, CPoint start\_p, CPoint end\_p, int color);

virtual void transform(int a[][3], int b[][3], int c[][3]);

在地信19-1班\_07192329\_4View.cpp中添加对应的函数定义，其中，中点画线函数前面已经定义，直接复制，不再列出。

void C地信191班071923294View::transform(int a[][3], int b[][3], int c[][3])

{

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

int sum = 0;

for (int k = 0; k < 3; k++)

{

sum += b[i][k]\*c[k][j];

}

a[i][j] = sum;

}

}

}

同时，OnDraw函数修改为：

void C地信191班071923294View::OnDraw(CDC\* pDC)

{

C地信191班071923294Doc\* pDoc = GetDocument();

ASSERT\_VALID(pDoc);

if (!pDoc)

return;

//绘制原三角形

CPoint p1(100, 300), p2(300, 300), p3(200, 100);

midpoint\_line(pDC, p1, p2, (0, 0, 0));

midpoint\_line(pDC, p2, p3, (0, 0, 0));

midpoint\_line(pDC, p3, p1, (0, 0, 0));

//定义变换矩阵

int trs1[3][3] = { {1,0,0},{0,1,0},{-300,-300,1} };//平移

int rot[3][3] = { {0,1,0},{-1,0,0},{0,0,1} };//旋转

int trs2[3][3]{ {1,0,0},{0,1,0},{300,300,1} };//平移

//点转换为矩阵

int mtx1[3][3] = { {200,100,1},{100,300,1},{300,300,1} };

int mtx2[3][3] = { {0,0,0},{0,0,0} ,{0,0,0} };

int mtx3[3][3] = { {0,0,0},{0,0,0} ,{0,0,0} };

int mtx4[3][3] = { {0,0,0},{0,0,0} ,{0,0,0} };

//实现坐标变换

transform(mtx2,mtx1, trs1);

transform(mtx3,mtx2, rot);

transform(mtx4,mtx3, trs2);

//生成新的点

CPoint p4(mtx4[0][0], mtx4[0][1]);

CPoint p5(mtx4[1][0], mtx4[1][1]);

CPoint p6(mtx4[2][0], mtx4[2][1]);

//绘制新三角形

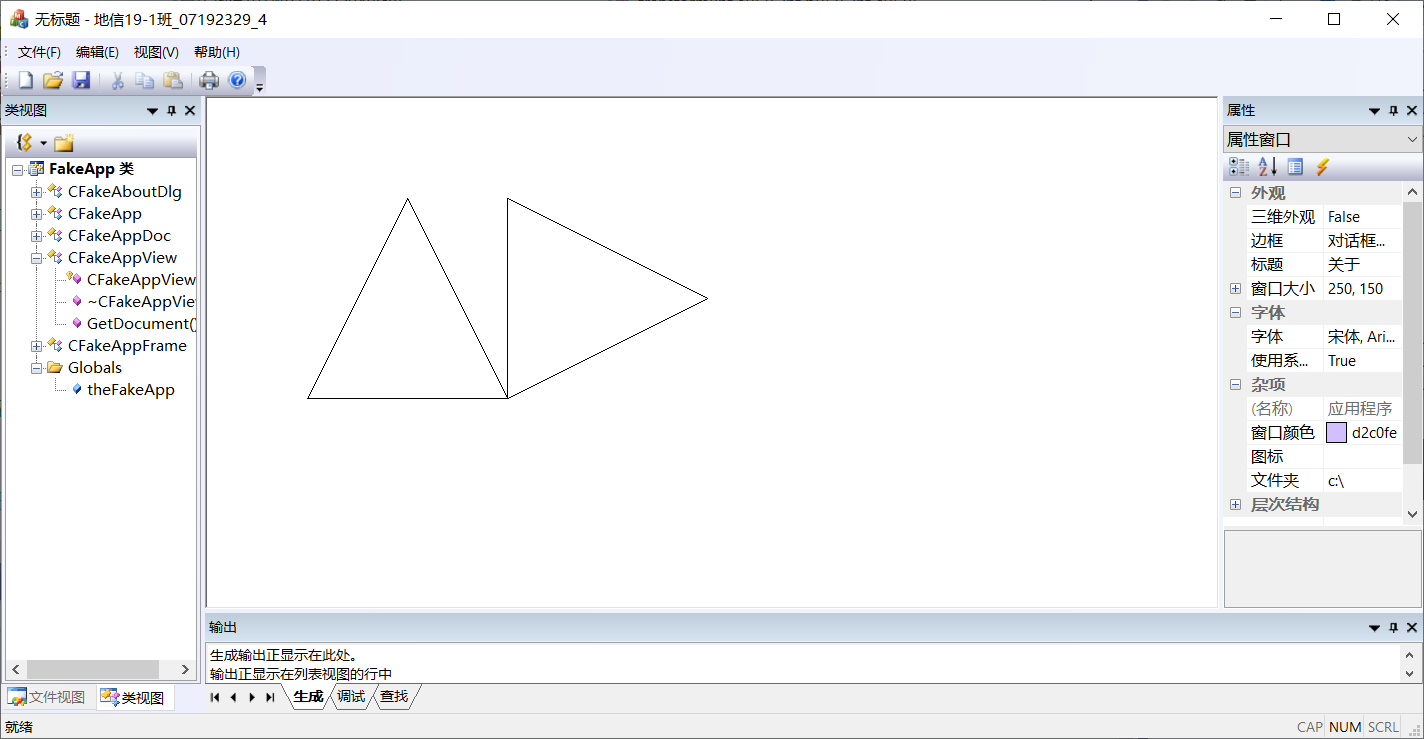
midpoint\_line(pDC, p4, p5, (0, 0, 0));

midpoint\_line(pDC, p5, p6, (0, 0, 0));

midpoint\_line(pDC, p6, p4, (0, 0, 0));

}

1. 实验结果



1. 实验体会

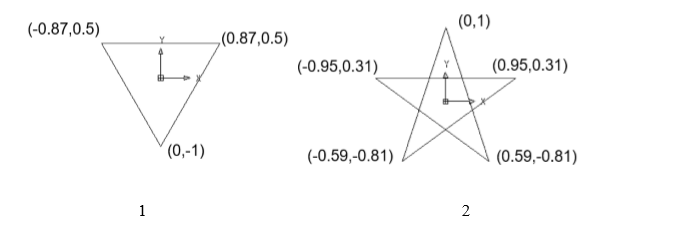
二维图形的基本变换，实质就是定义一个变换矩阵，并且为矩阵设置合适参数，再把二位图型的顶点构成的矩阵和变换矩阵相乘。我用了3\*3的二维数组，三重循环实现了相乘。要注意先初始化结果数组为{ {0,0,0},{0,0,0} ,{0,0,0} }。

实验五：点状地物的符号化

1. 实验目的

了解地图符号化基本原理，掌握点状地图符号的绘制方法。

1. 实验内容
2. 创建一个地图符号库，设计一个统一的地图符号数据结构，并在地图符 号库中添加以下两种地图符号的信息。



2）在实验二的基础上，根据地图符号库的地图符号以及 City 表的 Rank 属性，将不同城市以不同的地图符号绘制出来。

三、 实验思路

定义一个符号类，在cdc类中添加一个符号类的指针作为成员变量，同时定义一个readsym函数，从文件中读取每个符号的信息，利用这个函数初始化cdc中的符号类指针，使它指向一片存储有所有符号信息的一片连续的存储空间，以便在程序运行的开始就加载完所有符号。同时定义一个drawsym函数，把符号平移到指定位置，并进行适当缩放。在OnDraw函数中调用drawsym函数进行地物符号的绘制。

1. 关键代码

在实验二的基础上添加如下代码：

头文件：

void C地信191班071923292View::OnDraw(CDC\* pDC)

{

C地信191班071923292Doc\* pDoc = GetDocument();

ASSERT\_VALID(pDoc);

if (!pDoc)

return;

CPen mypen1(6, 2, RGB(0, 255, 0));

CPen\* oldpen = pDC->SelectObject(&mypen1);

for (int i = 0; i < chinaPtsCount - 1; i++)//绘制china

{

pDC->MoveTo(pChinaPts[i].x, 600 - pChinaPts[i].y);

pDC->LineTo(pChinaPts[i + 1].x, 600 - pChinaPts[i + 1].y);

}

pDC->SelectObject(oldpen);

read\_psym();

for (int i = 0; i < cityPtsCount; i++)//绘制city

{

//pDC->Ellipse(pCityPts[i].x - 2, 600 - pCityPts[i].y - 2, pCityPts[i].x + 2, 600 - pCityPts[i].y + 2);

////////////////////////////////////////////////////////////////////

//////字符串类型转换，将string转为CString

////////////////////////////////////////////////////////////////////

#ifdef \_UNICODE //如果是unicode工程

USES\_CONVERSION; CString tempStr(pCityPts[i].name.c\_str());

#else //如果是多字节工程

CString ans;

tempStr.Format("%s", pCityPts[i].name.c\_str());

#endif // \_UNICODE

////////////////////////////////////////////////////////////////////

pDC->TextOut(pCityPts[i].x + 3, 600 - pCityPts[i].y + 3, tempStr); //绘制注记

int flag = 0;

if (pCityPts[i].rank - 1)

flag = 1;

draw\_psym(pDC, psym[flag], pCityPts[i].x, pCityPts[i].y, 15);//绘制符号

}

}

void C地信191班071923292View::read\_psym()

{

fstream openpsym;

int sym\_count = 0;//符号数量

openpsym.open("地图符号库.txt", ios::in);//读入文件

openpsym >> sym\_count;//读入点数

if (psym) delete []psym;//清理内存

psym = new sym[sym\_count];//动态分配

for (int i = 0; i < sym\_count; i++)

{

openpsym >> psym[i].sym\_num;//读入编号

openpsym >> psym[i].p\_num;//读入点数

psym[i].px = new double[psym[i].p\_num];//动态分配x

psym[i].py = new double[psym[i].p\_num];//动态分配y

for (int j = 0; j < psym[i].p\_num; j++)

{//传入x坐标//传入y坐标

openpsym >> psym[i].px[j] >> psym[i].py[j];

}

}

openpsym.close();

}

void C地信191班071923292View::draw\_psym(CDC\* pDC, sym psym, float xpan, float ypan, float scale)

{

double\* ptr\_x = psym.px;//读取

double\* ptr\_y = psym.py;

for (int i = 0; i < psym.p\_num; i++)

{

pDC->MoveTo(ptr\_x[i] \* scale + xpan,

600 - (ptr\_y[i] \* scale + ypan));

pDC->LineTo(ptr\_x[i+1] \* scale + xpan,

600 - (ptr\_y[i+1] \* scale + ypan));

}

}

头文件：

struct MyPoint

{

int x;

int y;

int rank;

string name;

int id;

};

class sym

{

public:

int sym\_num; //符号编号

int p\_num;//点编号

double\* px;//x数组

double\* py;//y数组

};

public:

MyPoint\* pChinaPts;//存储China数据

MyPoint\* pCityPts;//存储City数据

int chinaPtsCount;//China的点数

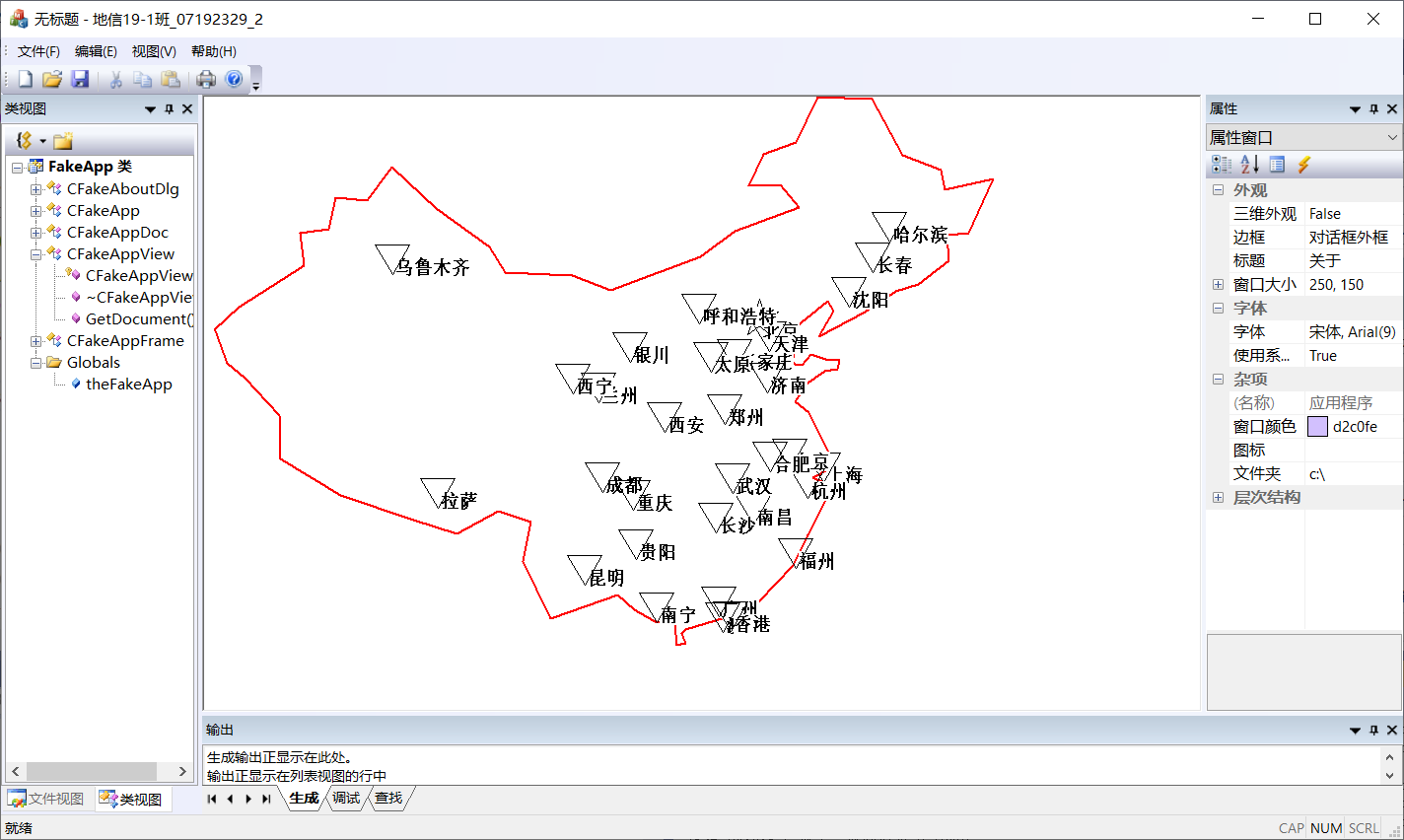
int cityPtsCount;//City 的点数

sym\* psym = NULL;

virtual void read\_psym();

virtual void draw\_psym(CDC\* pDC, sym psym, float xpan, float ypan, float scale);

1. 实验结果



1. 实验体会

将一个指向地图符号sym类的动态数组的指针\*psym作为CDC的成员变量，将指向存储坐标的动态数组的指针作为sym类的成员变量，通过这两个动态数组的“嵌套”，就可以方便的在调用CDC类的时候通过read\_psym函数读取地物符号的信息，并且把它全都引入CDC类。不过我使用文件流的时候遇到了不少问题，最后使用折中的办法解决了。实现了从TXT文件中读入符号信息并且调用绘制函数，要注意scale不能为0。