**1、加载头文件：**#include <graphics.h>

**2、基本图形函数：**

setcolor(RGB(r,g,b)); /\* 颜色设置 \*/

**putpixel(x, y, color); /\* 写点 \*/**

color=getpixel(x, y); /\* 读点 \*/

line(x1,y1,x2,y2); /\* 画线 \*/

circle(x1,y1,r); /\* 画圆 \*/

rectangle(x1,y1,x2,y2); /\* 画矩形 \*/

**3、DDA算法程序题：**

void DDALine(VPOINT p1, VPOINT p2, int color)

{ int delta, i; // delta为x、y方向最大位移的绝对值

float xinc, yinc;

float x，y; // 留心①：必须float类型

if(abs(p2.x-p1.x)>abs(p2.y-p1.y))

{ delta=abs(p2.x-p1.x);

if(p2.x-p1.x>0) xinc=1; else xinc=-1;

yinc=（float）(p2.y-p1.y)/abs(p2.x-p1.x);

}

else

{ delta=abs(p2.y-p1.y);

if(p2.y-p1.y>0) yinc=1; else yinc=-1;

xinc=（float）(p2.x-p1.x)/abs(p2.y-p1.y);

}

x=p1.x; y=p1.y;

for(i=1; i<=delta; i++)

{ putpixel(x+0.5, y+0.5, color);

x=x+xincre; y=y+yincre;

}

}

**4、字符和线型的实现**

#include <stdio.h>  
#include <graphics.h>  
#include <math.h>  
int linestyle[] ={1,1,1,1,1,0,0,0,0,0};  
typedef struct VPOINT{  
int x;  
int y;  
}VPOINT;  
void DDA(VPOINT p1,VPOINT p2){  
initgraph(640, 480);  
float incx;  
float incy;  
float delta;  
float x,y;  
if(abs(p2.y-p1.y)<=abs(p2.x-p1.x))**{**  
delta = abs(p2.x-p1.x);  
if(p2.x>p1.x) incx = 1;  
else incx = -1;  
incy = (float)(p2.y-p1.y)/(p2.x-p1.x);  
x = p1.x;  
y = p1.y;  
int i=0;  
for(;i<delta;i++){  
if(linestyle[i%10]==1)  
putpixel((int)x,(int)y,255);  
x+=incx;  
y+=incy;  
}  
}else{  
delta = abs(p2.y-p1.y);  
if(p2.y>p1.y) incy = 1;  
else incy = -1;  
incx = (float)(p2.x-p1.x)/(p2.y-p1.y);  
int i=0;  
x = p1.x;  
y = p1.y;  
for(;i<delta;i++){  
if(linestyle[i%10]==1)  
putpixel((int)x,(int)y,255);  
x+=incx;  
y+=incy;  
}  
}  
getchar();  
getchar();  
closegraph();  
}  
int main(){  
VPOINT a,b;  
scanf("%d %d %d %d",&a.x,&a.y,&b.x,&b.y);  
DDA(a,b);  
return 0;  
}

**5、简答题**

**（2）反采样方法：**

答：基本上反走样方法可分为两类。第一类是提高分辨率 即增加采样点(提高采样频率)，通过硬件技术，提高显示器的分辨率来实现；另一类反走样软件技术（加权区域采样）把像素作为一个有限区域，对区域采样来调整像素的亮度，以光顺边界来减小锯齿现象。

**（3）半色调技术定义：**

答：用较少的颜色来显示较多的亮度等级。

**（4）判断点与多边形的位置关系方法？**

答：叉积判断法、夹角之和检测法、交点技术检测法、判断多边形的凹凸性。

夹角之和检测法;

从作一射线至无穷远： 

计算射线与多边形边的交点个数。

若交点数为奇数，则在多边形内；

否则，在多边形外。

奇异点(多边形顶点)处理：

若相邻两边在射线同侧，则交点计数加2，否则加1。

**(5)请从除有序边表之外的其余关于多边形算法有哪些？**

答：边填充算法、边标志算法、种子填充算法

边标志算法步骤：

①对多边形的所有边用特殊颜色做标志

②inside初值为false，表示当前像素在多边形外部。

③对每条与多边形相交的扫描线，从左向右访问其中每个像素。每当遇到一个特殊颜色的像素，对inside取反。

④若inside==true，当前像素在多边形内部，着色。

**（6）种子填充算法（用递归实现）**

void FlodFill（int x，int y，int old，int fill）

｛ int current = getpixel（x,y）;

If(current==old)

{ FoldFill(x,y-1,old,fill);

FoldFill(x,y+1,old,fill);

FoldFill(x-1,y,old,fill);

FoldFill(x+1,y,old,fill);

}

｝

**(7)什么叫齐次坐标？使用齐次坐标有什么优点？**

答：齐次坐标就是n维空间中的物体可用n+1维坐标空间来表示。

优点：它提供了一个三维空间中包括平移、旋转、透视、投影、反射、错切、和比例在内的统一表达式，使得物体的变换可在同一的矩阵形式下进行运算，简化了计算量。

#### （8）多边形的一种更通用的存储结构

typedef struct **/\* 点集结构 \*/**

{ VPOINT \*points;

int Count;

}VPOINTS;

typedef struct **/\* 边结构 \*/**

{ int spID,epID;

}EDGE;

typedef struct **/\* 边集结构 \*/**

{ EDGE \*edges;

int Count;

}EDGES;

typedef struct **/\* 多边形结构 \*/**

{ VPOINTS points;

EDGES edges;

}VPOLYGON;

**/\* 绘制多边形 \*/**

void DrawPolygon(VPOLYGON polygon)

{ int i, spID, epID;

VPOINT sp, ep;

for(i=0; i<polygon.edges.Count ; i++)

{ spID=polygon.edges.edges[i].spID;

epID=polygon.edges.edges[i].epID;

sp=polygon.points.points[spID];

ep=polygon.points.points[epID];

line(sp.x, sp.y, ep.x, ep.y);

}

}

##### （9）1、水平面(XOY)投影





##### 2、正面(XOZ)投影

1. Y = 0；
2. 绕X轴正向旋转90°,使其与XOY面共面；
3. 沿Y方向平移。



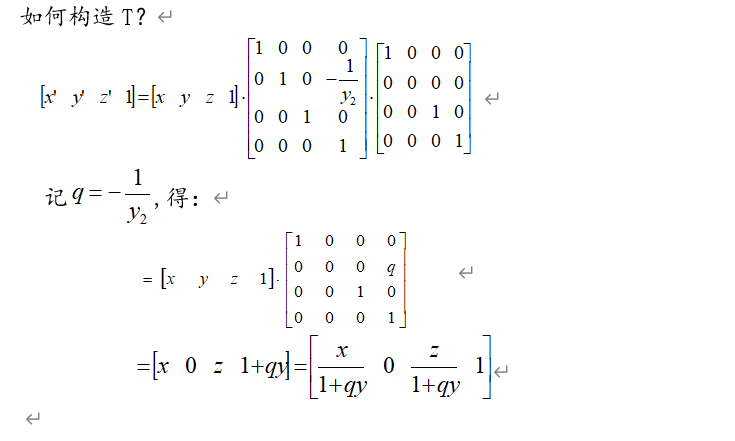
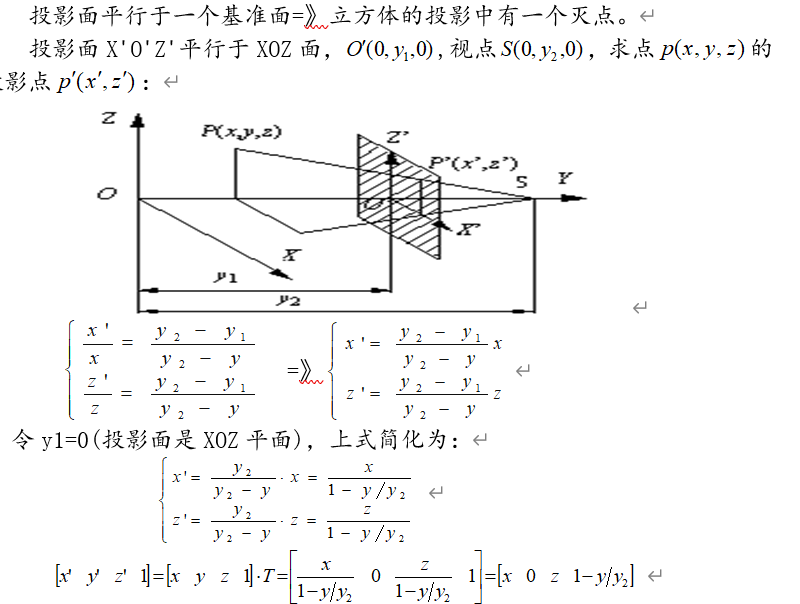


##### 3、侧面(YOZ)投影

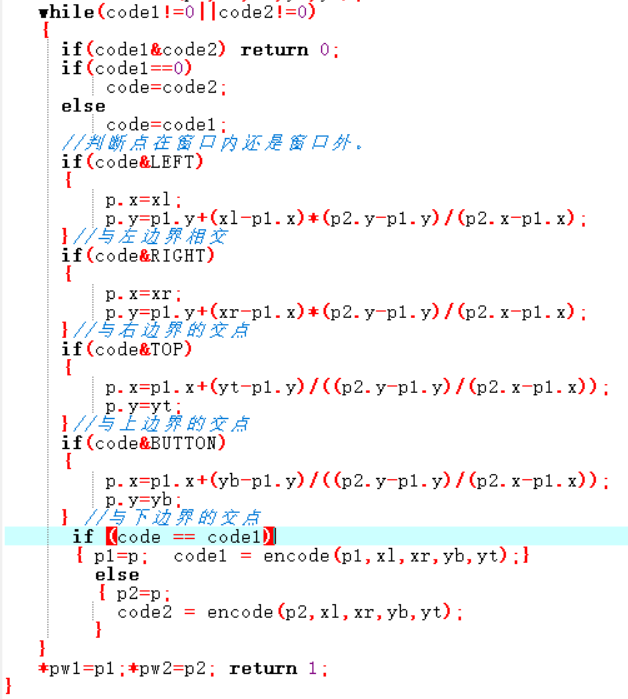
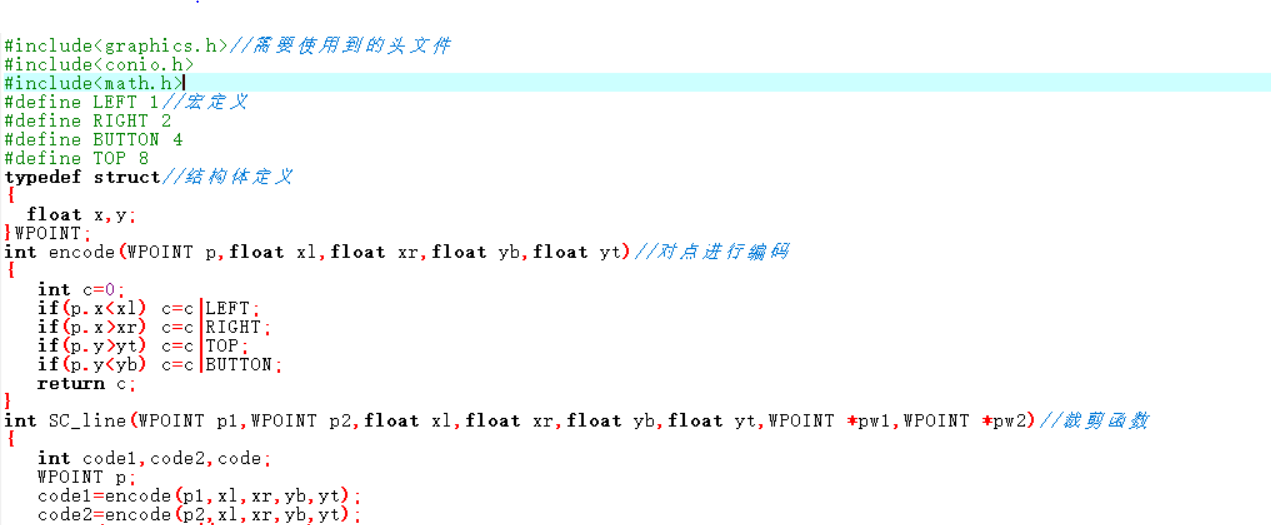
1. X=0；
2. 绕Y轴正向旋转-90°，使其与XOY面共面；
3. 沿X方向平移。

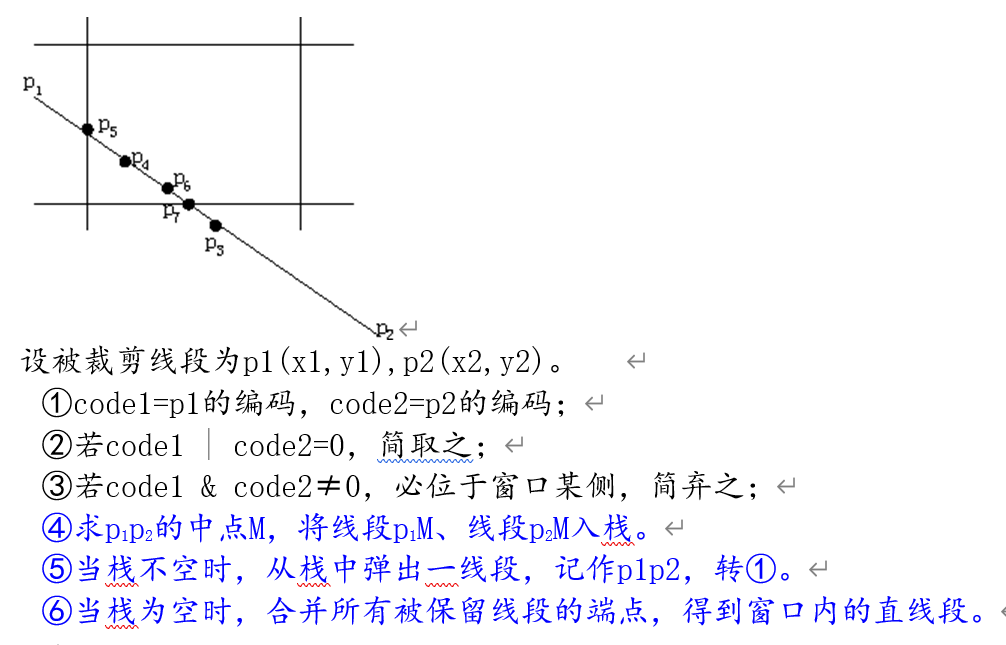
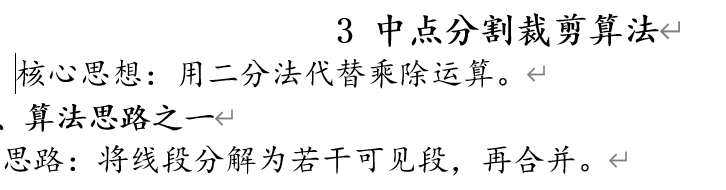
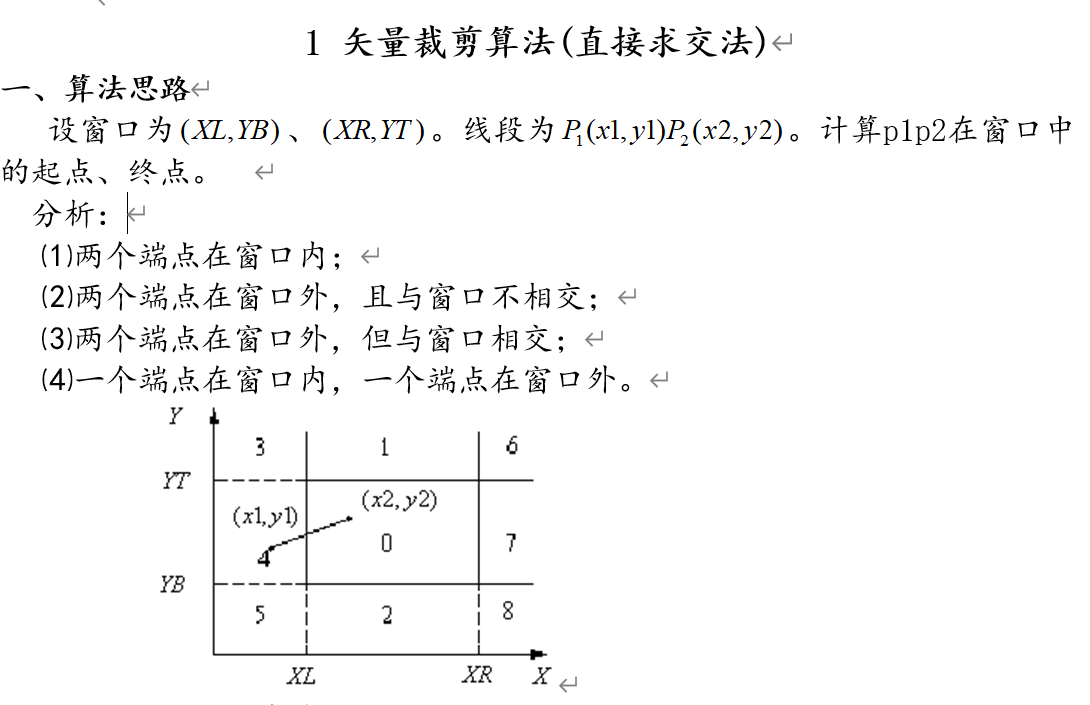




**（10）一点透视**

**（11）裁剪算法**





**六、计算题**

**（1）二维基本变换：平移、缩放、对称、旋转**

#### ①平移变换

相对坐标原点平移：

#### ②比例变换

相对坐标原点，沿x方向放缩a倍，沿y方向放缩d倍：

****

#### ③旋转变换

将P点绕坐标原点转动某个角度（逆时针为正）得到点P'。

绕坐标原点逆时针旋转θ角：****

#### ④对称变换

　（1）对x轴的对称变换： 

（2）对y轴的对称变换： 

　（3）对原点的对称变换： 

　（4）对y=x的对称变换：

（5）对y=-x的对称变换：

**（2）三维基本变换：平移、缩放、对称、旋转**

**（3）例：绕任一点逆时针旋转θ角，求变换矩阵。**

|  |  |  |
| --- | --- | --- |
| ① | 将原点平移到 |  |
| ② | 绕原点逆时针旋转θ角 |  |
| ③ | 将原点平移到 |  |
|  | 总变换矩阵 |  |

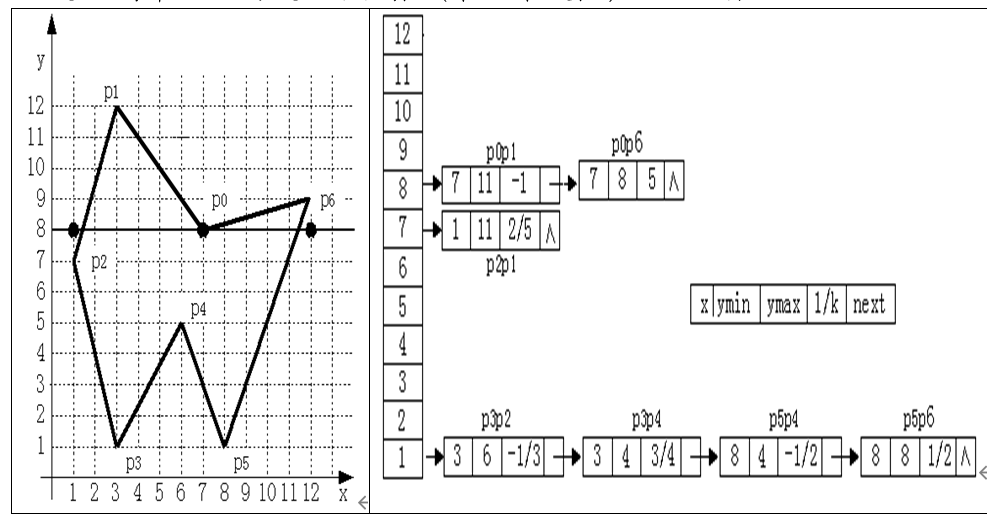
**（4）例：已知直线L：，求相对L的对称变换矩阵。**

由已知得：X轴截距=-C/A Y轴截距=-C/B

与X轴左侧夹角。

|  |  |  |
| --- | --- | --- |
| ① | 将原点平移到(-C/A,0)；  等价于点相对坐标原点平移(C/A,0) |  |
| ② | 坐标轴绕原点逆时针旋转θ，使X坐标轴与直线重合；  等价于点绕原点逆时针旋转-θ |  |
| ③ | 点对X坐标轴对称变换； |  |
| ④ | 坐标轴绕原点逆时针旋转-θ，坐标轴归位； |  |
| ⑤ | 将原点平移(C/A,0),原点归位。 |  |
|  | 总变换矩阵 |  |

（5）有序边表的数据结构：



#### （6）窗口-视区变换

设WCS和DCS的坐标轴平行。

窗口：左下角点，右上角点；

视区：左下角点，右上角点。

窗口内点投影到视区内点：



记： 



=》 

=》 