计算机图形学题库参考答案

一、全象限整型 DDA 直线段生成算法的 C 描述。

```
void DDAline(int x1, int y2, int x2, int y2)

{
  int I, steps;
  float x,y,dx,dy;
  x = x1+0.5; y = y1+0.5;
  steps = abs(x2-x1) > abs(y2-y1) ? abs(x2-x1) +1: abs(y2-y1)+1;
  dx = (float)(x2 - x1)/steps;
  dy = (float)(y2 - y1)/steps;
  for (I = 0; I < steps; I++) {
      putpixwl((int)x, (int)y, 15);
      x += dx; y += dy;
  }
}
```

二、全象限整型 Bresenham 直线段生成算法的 C 描述。

```
void Bline(int x1, int y1, int x2, int y2)
{
  int x, y, dx, dy, temp, xSign, ySign, interchange;
  interchange = 0;
  x = x1; y = y1;
  dx = abs(x2 - x1);
  dy = abs(y2 - y1);
  xSign = (x2 > x1) ? 1 : -1;
  ySign = (y2 > y1) ? 1 : -1;
  if (dy > dx) {
      interchange = 1;
      temp = dx; dx = dy; dy = temp;
  }
  e = 2*dy - dx;
  for ( i = 0; i < dx; i++) {
      putpixel(x, y, 15);
      if (e > 0 ) {
          if (interchange) x += xSign;
          else y += ySign;
          e = e - 2*dx;
      }
      if (interchange) y += ySign;
      else x += xSign;
      e = e + 2*dy;
  }
}
```

三、空间变换实现绕轴顺时针旋转一定角度。

$$T_1 = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x1 & -y1 & -z1 & 1 \end{vmatrix} \qquad T_2 = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & C/V & B/V & 0 \\ 0 & -B/V & C/V & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$T_3 = \begin{vmatrix} V & 0 & -C & 0 \\ 0 & 1 & 0 & 0 \\ C & 0 & V & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \qquad T_4 = \begin{vmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$\mathbf{T} = T_1 T_2 T_3 T_4 T_3^{-1} T_2^{-1} T_1^{-1}$ $\qquad V = \sqrt{B^2 + C^2}$

注: 答案不唯一, 可以有两种.

四、空间变换使某轴与坐标轴正向对齐。

1. 与 Y 轴正向对齐， Y 逆->Z 逆。

$$T_1 = \begin{vmatrix} A/V & 0 & -C/V & 0 \\ 0 & 1 & 0 & 0 \\ C/V & 0 & A/V & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \qquad T_2 = \begin{vmatrix} B & V & 0 & 0 \\ -V & B & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$V = \sqrt{A^2 + C^2}$$

$$T = T_1 * T_2$$

2. 与 X 轴正向对齐。

构造空间变换 T，使过原点的任意轴(A，B，C)与 X 轴正向对齐。

$$T_1 = \begin{vmatrix} C/V & -B/V & 0 & 0 \\ B/V & C/V & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \qquad T_2 = \begin{vmatrix} U & 0 & -C & 0 \\ 0 & 1 & 0 & 0 \\ C & 0 & U & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$V = \sqrt{B^2 + C^2} \qquad U = \sqrt{A^2 + B^2}$$

$$T = T_1 * T_2$$

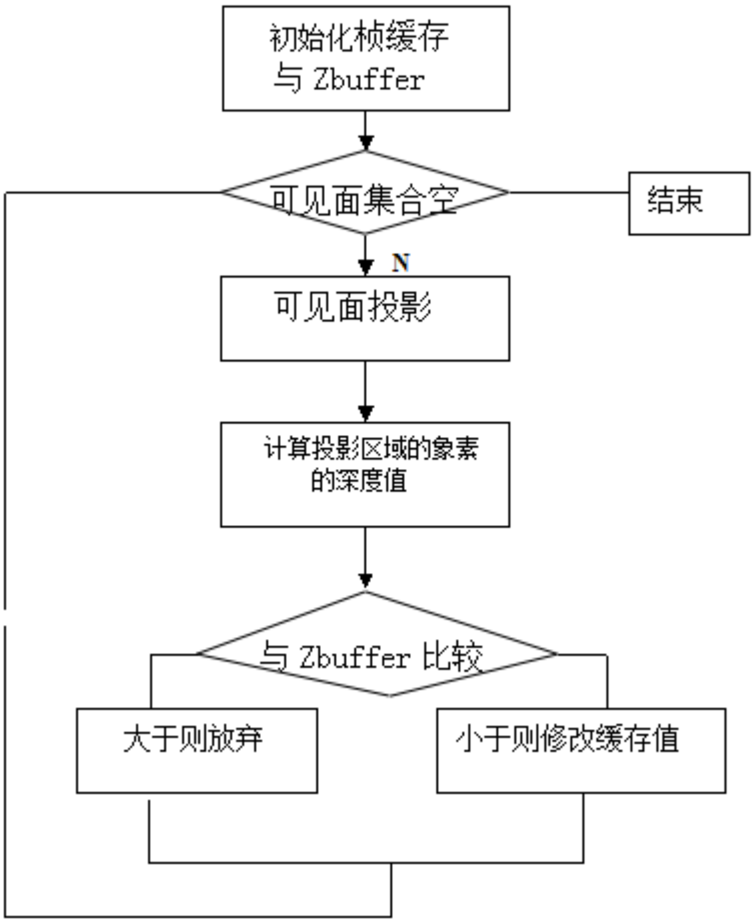
注: 答案不唯一, 可以有四种.

五、 Zbuffer 消隐原理及流程框图。

Zbuffer 消隐算法的基本原理:

与桢缓存对等建立深度缓存 Zbuffer，保存屏象素的深度值。空间可见面投影后，计算投影区域的每一象素的深度值，并与 Zbuffer 中的深度值进行比较，值小的则保存。

处理流程框图：

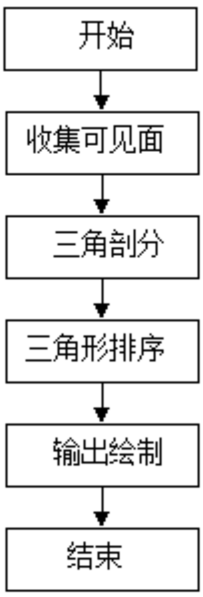


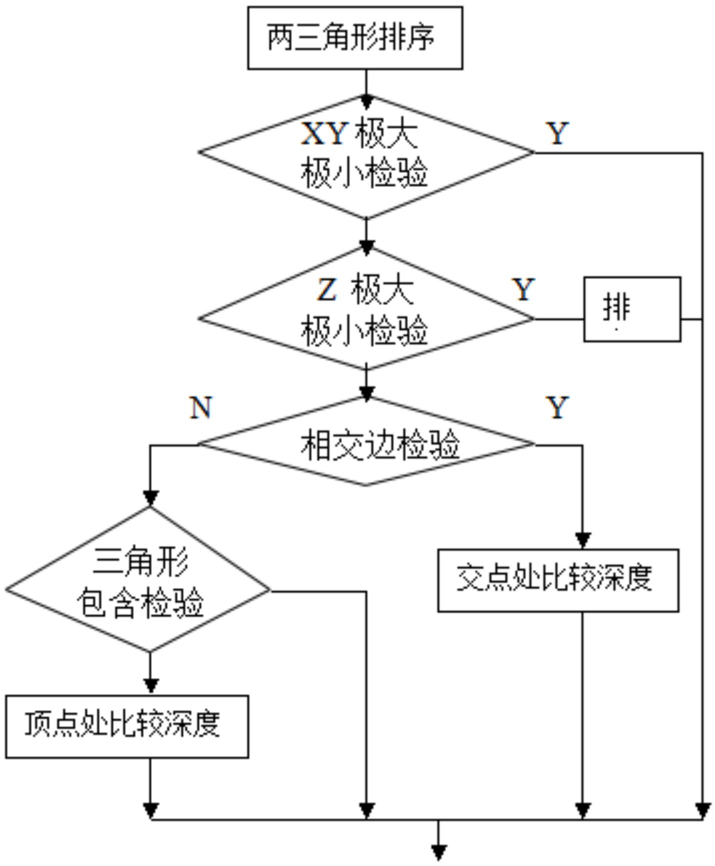
六、画家消隐原理及流程框图。

画家消隐算法的基本原理:

如同画家绘画一样，从远景开始逐步绘制到近景。这样后绘的近景部分就会覆盖先绘的远景部分，达到消隐的目的。

处理流程框图：





七、二维窗口端点编码裁剪的函数描述。及直线段完全可见、不可见、部分可见的充要条件。

端点编码设计方案:

采用四位编码，如下:

上	下	左	右

对应位置 0 为窗口边界可见一侧，1 不可见一侧。

1010	1000	1001
0010	0000	0001
0110	0100	0101

编码结果如上图。

```
int  pCode(int x, int y)
     {
        int code = 0;

          if  (x > Wrx) code |= 1;
          if  (x < Wlx) code |= 2;
          if  (y > Wry) code |= 4;
          if  (y < Wly) code |= 8;
          return code;
     }
```

直线段完全可见的充要条件: $!\,P_1code \;\&\&\; !\,P_2code$

完全不可见的充分条件: $P_1code \;\&\; P_2code$

部分可见的充分条件: $!\,P_1code \;\&\&\; P_2code \;||\; P_1code \;\&\&\; !P_2code$

八、4 控制点三次 Bezier 曲线上特定点的坐标。

$P(0) = P_1(0，0)$

$P(1/3) = (1210/27, 850/27)$

$P(1/2) = (62.5，38.5)$

$P(1) = P_4(110，10)$

注:计算过程略

九、特征多边形确定两段三次 Bezier 曲线在分段点达到 G^1 连续的解析条件和 C^1 连续的条件。

注:Bezier 曲线在起始点和结束点位置的一阶导数推导略。

根据三次 Bezier 曲线在起始点和结束点位置的一阶导数性质可知，两段曲线在分段点 $P_4(Q_1)$处达到 G^1 连续的条件是前后三点共线，既要求 Q_2 点应在 P_3 和 P_4 两点构成的直线的延长线上。解析条件为:

$$\begin{cases} x = 2t + 2 \\ y = t - 1 \qquad t > 1 \\ z = t - 1 \end{cases}$$

在分段点 $P_4(Q_1)$处达到 C^1 连续的条件是 G^1 连续条件中 $t = 1$。

十、空间透视中心到空间投影面透视裁剪约束体的上、下、左、右四个裁剪面的方程。

上: $Z = (Ze - Z_2) / Z * X + Z_2$

下: $Z = (Ze - Z_1) / Z * X + Z_1$

左: $Y = (Ye - Y_1) / X * X + Y_1$

右: $Y = (Ye - Y_2) / X * X + Y_2$

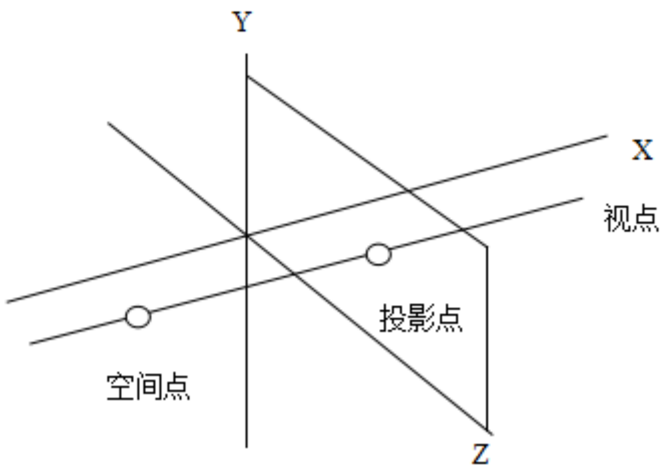
十一、三维图形处理流程。对空间投影做如下说明：建模方式，屏幕右下角投影窗口的实现步骤。

三维图形处理的一般流程：

建模、空间变换、背面移去、裁剪、投影、光照、绘制。

投影模型：

YOZ 坐标平面为空间投影面。建模方式如图：



空间投影窗口变换到屏幕界面右下角显示窗口的实现步骤：

1．投影窗口内投影点坐标归一化。

2．根据显示窗口尺寸对投影点归一化坐标进行缩放处理。

3．绘制。