实现思路

1. 三角形边框

画出三角形的三条边即可，而每条边可以使用 Bresenham 算法画直线。

实现函数：

```
void bresenham(int x0, int y0, int x1, int y1, float points[], int& len) {
    bool steep = abs(y1 - y0) > abs(x1 - x0);
    if (steep) {
        swap(x0, y0);
        swap(x1, y1);
    }
    if (x0 > x1) {
        swap(x0, x1);
        swap(y0, y1);
    }
    int dx = x1 - x0;
    int dy = abs(y1 - y0);
    int dy2 = 2 * dy;
    int dy2minusdx2 = dy2 - 2 * dx;
    int p = dy2 - dx;

    points[len++] = steep ? y0 : x0;
    points[len++] = steep ? x0 : y0;
    int y = y0;
    int ystep = y0 < y1 ? 1 : -1;
    for (int x = x0 + 1; x <= x1; x++) {
        if (p <= 0) {
            p += dy2;
        }
        else {
            y += ystep;
            p += dy2minusdx2;
        }
        points[len++] = steep ? y : x;
        points[len++] = steep ? x : y;
    }
}
```

通过检查dx和dy决定从x轴还是y轴进行遍历；通过判断x0和x1的大小将小的点放在前面。之后使用Bresenham 算法设置p的初始值，根据p是否大于0决定p如何更新以及y是否前进。将得到的坐标存入points数组中即可。

2.圆

使用 Bresenham 算法和八分法得到圆的各个点。

实现函数：

```
void bresenham(int r, float points[], int& len) {
    int x = 0;
    int y = r;
    int p = 3 - 2 * r;
    for (x = 0; x <= y; x++) {
        points[len++] = x;
        points[len++] = y;

        points[len++] = -x;
        points[len++] = y;

        points[len++] = x;
        points[len++] = -y;

        points[len++] = -x;
        points[len++] = -y;

        points[len++] = y;
        points[len++] = x;

        points[len++] = -y;
        points[len++] = x;

        points[len++] = y;
        points[len++] = -x;

        points[len++] = -y;
        points[len++] = -x;

        if (p <= 0) {
            p += 4 * x + 6;
        }
        else {
            p += 4 * (x - y) + 10;
            y--;
        }
    }
}
```

由于圆的对称性，应用八分法，我们只需要得到从圆正上方开始八分之一圆的各点坐标即可得到整个圆环的各点坐标。对于这八分之一圆（圆心为原点），我们可以使用Bresenham 算法，由于这部分圆弧的斜率绝对值小于1，因此遍历x轴，从x = 0， y = r开始，判断下一个点应该为（x + 1， y）还是 （x + 1， y - 1），根据公式推导可以定义一个变量p，根据p是否大于0来更新p和选择下一点，直到x > y 为止。将得到的点的坐标存入points数组中即可。

画图思路

可以使用两个VOA和VBO对象，一个画网格，一个画点。每轮循环渲染中，先绑定网格的VOA绘制网格，再绑定点的VOA和VBO，调用 bresenham 函数得到新的各点坐标，将各点坐标归一化至[-1, 1]后用其更新VBO中存储的数据，链接顶点属性，绘制出三角形或圆的边框。