HW1 Ellipse

姓名：陈昱静
学号：3210106348
专业：计算机科学与技术

实验目的与要求

进行椭圆的扫描转换。

输入：椭圆中心 (x, y)；长轴 Ra；短轴 Rb；旋转角度 θ

输出：ppm 格式的椭圆图像

实验原理与内容

椭圆方程

椭圆方程的基本形式：$f(x,y) = Ax^2 + Bxy + Cy^2 + Ex + Fy + D = 0$
简单起见，此处我们假设中心 (x_c, y_c) 位于原点 $(0,0)$，则方程为：

$$f(x,y) = Ax^2 + Bxy + Cy^2 + D = 0$$

记一焦点坐标为 (x_f, y_f)，长轴为 a，短轴为 b，焦距为 c，椭圆逆时针旋转角度为 θ ，则有：

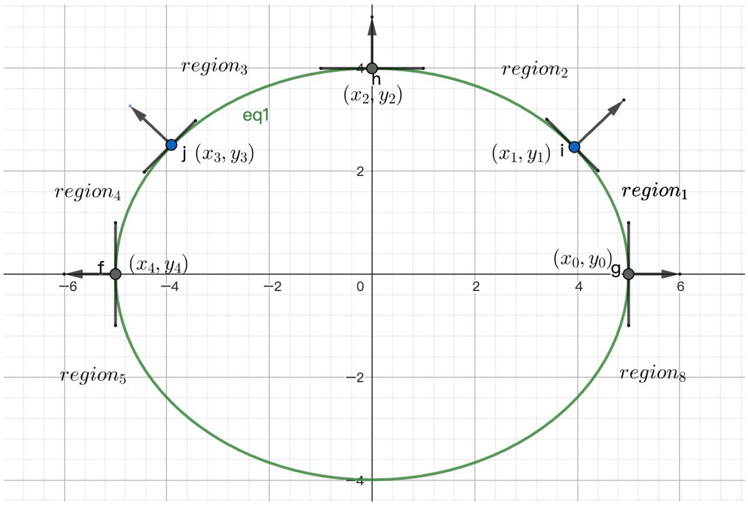
$$\begin{cases} c = \sqrt{a^2 - b^2} \\ x_f = c\cos\theta \\ y_f = c\sin\theta \end{cases} \qquad \begin{cases} A = a^2 - x_f^2 \\ B = -2x_f y_f \\ C = a^2 - y_f^2 \\ D = a^2(x_f^2 + y_f^2 - a^2) \end{cases}$$

区域划分

椭圆关于其长轴对称，故只需绘制其中一部分，另一部分由对称关系可得。此处不妨选择：

$$\begin{cases} Ax^2 + Bxy + Cy^2 + D = 0 \\ \cos\theta \cdot y - sin\theta \cdot x > 0 \end{cases}$$

本程序采用 Bresenham 算法，依据椭圆上每一点处的斜率 k 将我们选择的半个椭圆分为四个区域：$k_0 \in (-\infty, 1]$，$k_1 \in (-1, 0]$，$k_2 \in (0, 1]$，$k_3 \in [1, \infty)$。



椭圆上每一点处法向量 $\vec{n}$ 垂直于切线，故可转化为求法线方向。

区域 8 - 1 分界：$\vec{n_0}=(1,0)$；区域 1 - 2 分界：$\vec{n_1}=(\frac{\sqrt{2}}{2},\frac{\sqrt{2}}{2})$；区域 2 - 3 分界：$\vec{n_2}=(0,1)$；区域 3 - 4 分界：$\vec{n_3}=(-\frac{\sqrt{2}}{2},\frac{\sqrt{2}}{2})$；区域 4 - 5 分界：$\vec{n_4}=(-1,0)$。

法向量可由梯度求得：$\vec{n}=\frac{\vec{grad}(f(x,y))}{||\vec{grad}(f(x,y))||}=\frac{(2Ax+By,Bx+2Cy)}{||(2Ax+By,Bx+2Cy)||}$。实际在判断时，只要判断 $\vec{grad}f(x,y)=(2Ax+By,Bx+2Cy)$ 两个分量的正负以及大小关系就够了。

在实际运算时，为了避免每次都要将点代入计算，我们可以算出分界点的坐标，之后每次循环只要判断是否达到分界点就好了。

$$\begin{cases}Bx+2Cy=0\\Ax^2+Bxy+Cy^2+D=1\\x>0\end{cases}\Rightarrow\begin{cases}k_0=-\frac{B}{2C}\\x_0=\sqrt{\frac{-D}{A+Bk_1+Ck_1^2}}\\y_0=k_0x_0\end{cases}$$

$$\begin{cases}2Ax+By=Bx+2Cy\\Ax^2+Bxy+Cy^2+D=1\end{cases}\Rightarrow\begin{cases}k_1=-\frac{2A-B}{2C-B}\\x_1=\pm\sqrt{\frac{-D}{A+Bk_1+Ck_1^2}}\\y_1=k_2x_2\end{cases}$$

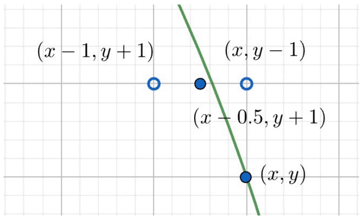
$$\begin{cases}2Ax+By=0\\Ax^2+Bxy+Cy^2+D=1\end{cases}\Rightarrow\begin{cases}k_2=-\frac{B}{2A}\\y_2=\sqrt{\frac{-D}{Ak_3^2+Bk_3+C}}\\x_2=k_3y_3\end{cases}$$

$$\begin{cases}2Ax+By=-(Bx+2Cy)\\Ax^2+Bxy+Cy^2+D=1\end{cases}\Rightarrow\begin{cases}k_3=-\frac{2A+B}{2C+B}\\x_3=\sqrt{\frac{-D}{A+Bk_4+Ck_4^2}}\\y_3=k_4x_4\end{cases}$$

由于分界点按逆时针顺序排列，我们每次判断 $(x_1, y_1) \times (x_i, y_i) = x_1 y_i - x_i y_1$ 的正负，若为负则取 (x_i, y_i) 关于原点的对称点作为分界点。

Bresenham 算法

以区域 1 为例。



当前所在点 (x, y)，下一个点选择为 $(x-1, y+1)$ 或 $(x, y+1)$。根据其中点 $(x-0.5, y+1)$ 位置进行选择。若中点在椭圆内部，即 $f(x-0.5, y+1) < 0$，则下一个点为 $(x, y+1)$，否则选择 $(x-1, y+1)$。

记 $p^{(1)} = f(x-0.5, y+1) = A(x-0.5)^2 + B(x-0.5)(y+1) + C(y+1)^2$，则每次只需要判断 p 的正负。下构造递推关系简化运算：

$$\begin{cases} \Delta {p_n}^{(1)}(x, y) \triangleq p^{(1)}(x, y+1) - p^{(1)}(x, y) = Bx + 2Cy + C \\ \Delta {p_{nw}}^{(1)}(x, y) \triangleq p^{(1)}(x-1, y+1) - p^{(1)}(x, y) = \Delta {p_n}^{(1)} - 2Ax - By + A - B \end{cases}$$

$$\begin{cases} \Delta p_{n\_n} \triangleq \Delta p_n^{(1)}(x, y+1) - \Delta {p_n}^{(1)}(x, y) = 2C \\ \Delta p_{n\_nw} = \Delta p_n^{(1)}(x-1, y+1) - \Delta {p_n}^{(1)}(x, y) = 2C - B \\ \Delta p_{nw\_n} \triangleq \Delta {p_{nw}}^{(1)}(x, y+1) - \Delta {p_{nw}}^{(1)}(x, y) = 2C - B \\ \Delta p_{nw\_nw} \triangleq \Delta {p_{nw}}^{(1)}(x-1, y+1) - \Delta {p_{nw}}^{(1)}(x, y) = 2(A - B + C) \end{cases}$$

$$(p_{i+1}^{(1)},\ \Delta {p_n}^{(1)},\ \Delta {p_{nw}}^{(1)}) = \begin{cases} (p_i^{(1)} + \Delta {p_n}^{(1)},\ \Delta {p_n}^{(1)} + \Delta p_{n\_n}, \Delta {p_{nw}}^{(1)} + \Delta p_{nw\_n}) & p_i^{(1)} < 0 \\ (p_i^{(1)} + \Delta {p_{nw}}^{(1)},\ \Delta {p_n}^{(1)} + \Delta p_{n\_nw}, \Delta {p_{nw}}^{(1)} + \Delta p_{nw\_nw}) & p_i \geq 0 \end{cases}$$

边界条件：$y < y_1$，当 $= y_1$ 时进入区域 2。注意原先我们有 $p^{(1)}(x, y) = f(x-0.5, y+1)$，进入区域 2 后，应变为 $p^{(2)}(x, y) = f(x-1, y+0.5)$，易知 $p^{(2)}(x, y) = p^{(1)}(x-0.5, y-0.5)$。同理有 $\Delta p_{nw}^{(2)}(x, y) = \Delta p_{nw}(x-0.5, y-0.5)$，$\Delta {p_w}^{(1)} \triangleq p(x-1, y) - p(x, y)$，$\Delta {p_w}^{(2)} = \Delta p_w(x-0.5, y-0.5)$。可求得：

$$\begin{cases}\Delta p_w{}^{(2)}(x,y) = -2Ax - By + 2A + \frac{1}{2}B = \Delta p_{nw}^{(1)} - \Delta p_n^{(1)} + A + \frac{3}{2}B \\ \Delta p_{nw}{}^{(2)}(x,y) = \Delta p_{nw}{}^{(1)} + A - C\end{cases}$$

$$\begin{cases}\Delta p_{w\_w} \triangleq \Delta p_w{}^{(2)}(x-1,y) - \Delta p_w{}^{(2)}(x,y) = 2A \\ \Delta p_{w\_nw} \triangleq \Delta p_w{}^{(2)}(x-1,y+1) - \Delta p_w{}^{(2)}(x,y) = 2A - B \\ \Delta p_{nw\_w} \triangleq \Delta p_{nw}{}^{(2)}(x-1,y) - \Delta p_{mw}{}^{(2)}(x,y) = 2A - B \\ \Delta p_{nw\_nw} = 2(A - B + C)\end{cases}$$

$$(p_{i+1}^{(2)}, \Delta p_w^{(2)}, \Delta p_{nw}{}^{(2)}) = \begin{cases}(p_i^{(2)} + \Delta p_{nw}, \Delta p_w^{(2)} + \Delta p_{w\_nw}, \Delta p_{nw}{}^{(2)} + \Delta p_{nw\_nw}) & p_i^{(2)} < 0 \\ (p_i^{(2)} + \Delta p_w, \Delta p_w^{(2)} + \Delta p_{w\_ww}, \Delta p_{nw}{}^{(2)} + \Delta p_{nw\_w}) & p_i^{(2)} \geq 0\end{cases}$$

边界条件：$x > x_2$。区域 3:

$$\begin{cases}\Delta p_w{}^{(3)}(x,y) = \Delta p_w{}^{(2)}(x,y-1) = \Delta p_w{}^{(2)} + B \\ \Delta p_{sw}{}^{(3)}(x,y) = p^{(1)}((x-0.5)-1,(y-1.5)-1) - p^{(1)}(x-0.5,y-1.5) = 2\Delta p_w{}^{(2)} - \Delta p_{nw}{}^{(2)} + 4C - B\end{cases}$$

$$\begin{cases}\Delta p_{w\_w} = 2A \\ \Delta p_{w\_sw} = 2A + B \\ \Delta p_{sw\_w} = 2A + B \\ \Delta p_{sw\_sw} = 2(A + B + C)\end{cases}$$

$$(p_{i+1}^{(3)}, \Delta p_w{}^{(3)}, \Delta p_{sw}{}^{(3)}) = \begin{cases}(p_i^{(3)} + \Delta p_w{}^{(3)}, \Delta p_w{}^{(3)} + \Delta p_{w\_w}, \Delta p_{sw}{}^{(3)} + \Delta p_{sw\_w}) & p_i < 0 \\ (p_i^{(3)} + \Delta p_{sw}{}^{(3)}, \Delta p_w{}^{(3)} + \Delta p_{w\_sw}, \Delta p_{sw}{}^{(3)} + \Delta p_{sw\_sw}) & p_i \geq 0\end{cases}$$

边界条件：$x > x_3$。区域 4:

$$\begin{cases}\Delta p_{sw}{}^{(4)} = \Delta p_{sw}{}^{(3)} + C - A \\ \Delta p_s{}^{(4)} = \Delta p_{sw}{}^{(3)} - \Delta p_w{}^{(3)} - B\end{cases}$$

$$\begin{cases}\Delta p_{sw\_s} = 2C + B \\ \Delta p_{sw\_sw} = 2(A + B + C) \\ \Delta p_{s\_s} = 2C \\ \Delta p_{s\_sw} = 2C + B\end{cases}$$

$$(p_{i+1}^{(4)}, \Delta p_{sw}^{(4)}, \Delta p_s{}^{(4)}) = \begin{cases}(p_{i+1}^{(4)} + \Delta p_{sw}{}^{(4)}, \Delta p_{sw}^{(4)} + \Delta p_{sw\_sw}, \Delta p_s{}^{(4)} + \Delta p_{s\_sw}) & p_i^{(4)} < 0 \\ (p_{i+1}^{(4)} + \Delta p_w{}^{(4)}, \Delta p_{sw}^{(4)} + \Delta p_{sw\_w}, \Delta p_s{}^{(4)} + \Delta p_{s\_w}) & p_i^{(4)} \geq 0\end{cases}$$

Clipping

本实验中，在绘制时，测试坐标是否超出视窗边界，若超出则不进行绘制。

实验步骤与分析

main

从输入读入椭圆中心 (x, y)，长轴 Ra，短轴 Rb，以及旋转角度 θ（正方向为逆时针方向）。

此处读入的坐标以窗口中心为原点，如 $(0, 0)$ 表示椭圆会出现在视窗的正中心。但在进行计算时，为方便起见像素坐标以左下角为原点，故进行坐标转换。

此处设置窗口大小为 400 * 400

```
int main()
{
  int x, y, ra, rb;
  double theta;
  // Get ra, rb, x, y, and theta from input
  // ......

  // the input (x, y) takes the center of the window as origin
  // transform it to the viewport coordinate
  // [-WIDTH/2, WIDTH/2 - 1] * [-HEIGHT/2, HEIGHT/2 - 1] -> [0, WIDTH - 1] *
[0, HEIGHT - 1]
  x += WIDTH / 2;
  y += HEIGHT / 2;

  // output buffer
  unsigned char data[HEIGHT*WIDTH*3];
  memset(data, 0, WIDTH * HEIGHT * 3);
  // draw the ellipse
  ellipse(x, y, ra, rb, theta, data);
  // output
  ppmWrite("result.ppm", data, WIDTH, HEIGHT);
  return 0;
}
```

ellipse

首先判断输入 Ra 是否大于等于 Rb。若发现 $Ra < Rb$，本程序不报错，选择绘制旋转 90 度的椭圆，以 Rb 为长轴，以 Ra 为短轴。

接着根据实验原理中列出的公式，计算各参数，并依次进行绘制。

```
// if a < b, set b to be the major axis
  if(a < b)
  {
    std::swap(a, b);
    theta -= PI / 2;
  }
  // focal length
  double c = sqrt(a * a - b * b);
  // focus (xf, yf)
  double xf = c * cos(theta);
  double yf = c * sin(theta);

  // general formula of ellipse (suppose the center is at (0, 0))
  // Ax^2 + Bxy + Cy^2 + D = 0
  double A = a * a - xf * xf;
  double B = -2 * xf * yf;
  double C = a * a - yf * yf;
  double D = a * a * (yf * yf - A);

  // boundary point of area_8 and area_0
  double k1 = - B / (2 * C);
  double x1 = sqrt(-D / (A + B * k1 + C * k1 * k1));
  double y1 = k1 * x1;
  x1 = round(x1); y1 = round(y1);

  // boundary point of area_0 and area_1
  double k2 = (2 * A - B) / (2 * C - B);
  double x2 = sqrt(- D / (A + B * k2 + C * k2 * k2));
  double y2 = k2 * x2;
  if(x1 * y2 - x2 * y1 < 0)
  {
```

```
    y2 = -y2; x2 = -x2;
  }
  x2 = round(x2); y2 = round(y2);

  // boundary point of area_1 and area_2
  double k3 = -B / (2 * A);
  double y3 = sqrt(-D / (A * k3 * k3 + B * k3 + C));
  double x3 = k3 * y3;
  if(x1 * y3 - x3 * y1 < 0)
  {
    x3 = -x3; y3 = -y3;
  }
  x3 = round(x3); y3 = round(y3);

  // boundary point of area_2 and area_3
  double k4 = -(2 * A + B) / (2 * C + B);
  double x4 = -sqrt(-D / (A + B * k4 + C * k4 * k4));
  double y4 = k4 * x4;
  if(x1 * y4 - x4 * y1 < 0)
  {
    y4 = -y4; x4 = -x4;
  }
  x4 = round(x4); y4 = round(y4);

  // begin point
  int x_pos = x1, y_pos = y1;
  int x_mid = x_pos - 0.5, y_mid = y_pos + 1;

  double dpn = B * x_mid + 2 * C * y_mid + C;
  double dpnw = dpn - 2 * A * x_mid - B * y_mid + A - B;
  double dpn_n = 2 * C;
  double dpn_nw = 2 * C - B;
  double dpnw_n = dpn_nw;
  double dpnw_nw = 2 * (A - B + C);

  // region 0
  double p = A * x_mid * x_mid + B * x_mid * y_mid + C * y_mid * y_mid + D;
  while(y_pos < y2)
  {
```

```
        draw(x_pos, y_pos, xc, yc, data);
        y_pos++;
        if(p < 0)
        {
          p += dpn;
          dpn += dpn_n; dpnw += dpnw_n;
        }
        else
        {
          x_pos -= 1;
          p += dpnw;
          dpn += dpn_nw;  dpnw += dpnw_nw;
        }
      }
    // ......
```

在绘制图像时，同时绘制关于椭圆长轴对称的两个点。由于图像是正立输出的，还要再进行一次坐标转换，将纵坐标进行倒置。

```
void draw(int x, int y, int xc, int yc, unsigned char* data)
{
  if(!isOutofWindow(x, y, xc, yc))
    data[(HEIGHT - (y + yc)) * WIDTH * 3 + (x + xc) * 3] = 255;
  if(!isOutofWindow(-x, -y, xc, yc))
    data[(HEIGHT - (-y + yc)) * WIDTH * 3 + (-x + xc) * 3] = 255;
}
```

坐标范围判断

判断 $(x + x_c, y + y_c)$ 是否属于 $[0, WIDTH - 1] \times [0, HEIGHT - 1]$ 即可

```
bool isOutofWindow(int x, int y, int xc, int yc)
{
  return (y + yc < 0) || (y + yc >= HEIGHT) || (x + xc < 0) || (x + xc >=
WIDTH);
}
```

实验环境及运行方法

编程语言：c++

c++ 版本：c++11

编译及运行：

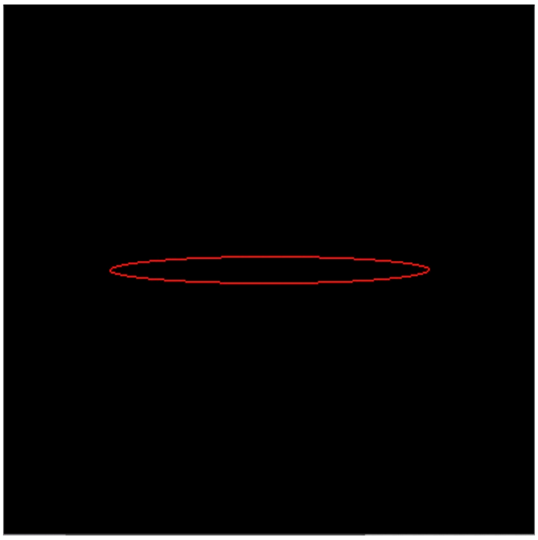
```
g++ -std=c++11 ellipse.cpp
./a.out
```

根据提示输入对应参数：

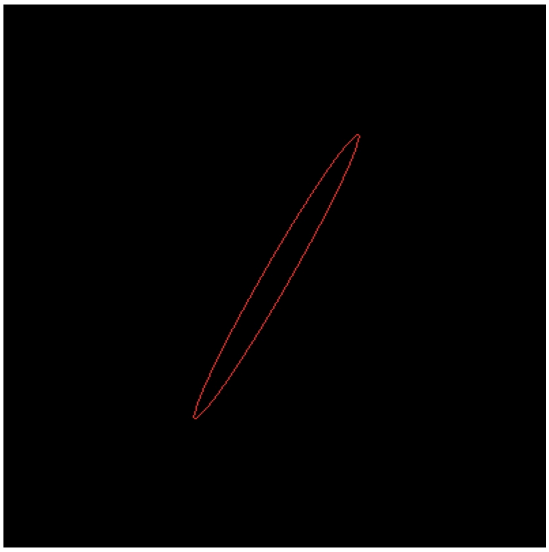
```
please input the semi-major axis a: [your input]
please input the semi-minor axis b: [your input]
please input the center
x: [your input]
y: [your input]
please input the rotation angle: [your input]
```

实验结果展示

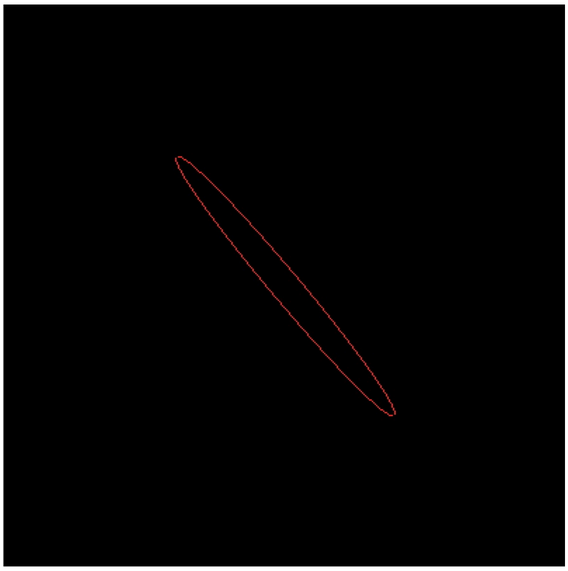
$Ra = 120, Rb = 10, (x, y) = (0, 0), \theta = 0^\circ$



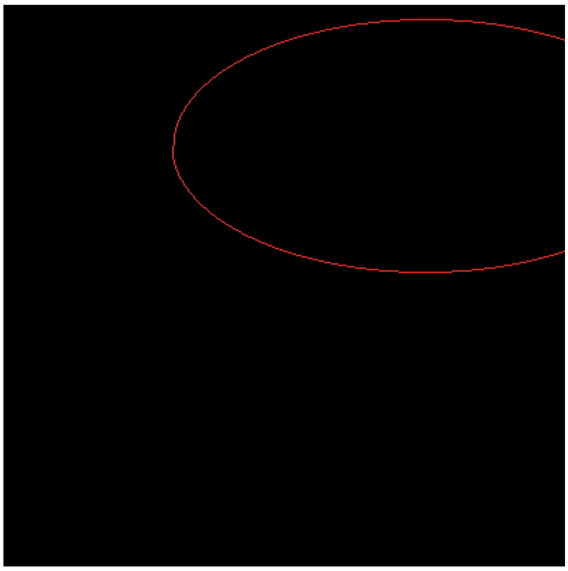
$Ra = 120, Rb = 10, (x, y) = (0, 0), \theta = 60^\circ$



$Ra = 120, Rb = 10, (x, y) = (0, 0), \theta = 490^\circ (= 130^\circ)$



$Ra = 180, Rb = 90, (x, y) = (100, 100), \theta = 0^\circ$



$Ra = 90, Rb = 180, (x, y) = (0, 0), \theta = 0^\circ$