```cpp
#include <iostream>

#include <cppqt.h>

using namespace std;
//#######################################################################
// malt vier Punkte oder zwei Linien
void drawEllipsePoints(Drawing& pic, int x, int y, int xcenter, int ycenter,
                       bool filled, int colour = 0)
{
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
  if (!filled)
    {
      pic.drawPoint(-x + xcenter,  y + ycenter, colour, true);
      pic.drawPoint( x + xcenter,  y + ycenter, colour, true);
      pic.drawPoint(-x + xcenter, -y + ycenter, colour, true);
      pic.drawPoint( x + xcenter, -y + ycenter, colour, true);
    }
  else
    {
      pic.drawLine(-x + xcenter,  y + ycenter, x + xcenter,  y + ycenter,
                   colour, true);
      pic.drawLine(-x + xcenter, -y + ycenter, x + xcenter, -y + ycenter,
                   colour, true);
    }
//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
}


//#######################################################################
// Scan Conversion für Ellipse, reelle Rechnung
```

```cpp
void drawEllipse_float(Drawing& pic, IPoint2D center, int a, int b, bool filled,
                       int colour = 0)
{
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
  int x     = 0;
  int y     = b;
  double a2 = a * a;
  double b2 = b * b;
  double d  = b - b2 / a2 - 0.25; //(dx0)

  drawEllipsePoints(pic, x, y, center.x, center.y, filled, colour);

  while (a2 * y > b2 * x)
    {
      if (d < 0) //S0
        {
          --y;
          d += 2 * y - b2 / a2 * (2 * x + 3);
        }
      else
        d -= b2 / a2 * (2 * x + 3);    //0
      ++x;
      drawEllipsePoints(pic, x, y, center.x, center.y, filled, colour);
    }

  d = a2 - a2/b2*y*y - x*x - x + a2/b2*(2*y-1) - 0.25;

  while (a2 * y > b2 * x)
    {
      if (d < 0)//S
```

```
          d += a2 / b2 * (2 * y - 3);
      else//S0
        {
          ++x;
          d -= 2 * x - a2 / b2 * (2 * y - 3);
        }
      --y;
      drawEllipsePoints(pic, x, y, center.x, center.y, filled, colour);
    }
//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
}

//#########################################################################
// Scan Conversion für Ellipse, ganzzahlige Rechnung
void drawEllipse(Drawing& pic, IPoint2D center, int a, int b, bool filled,
                 int colour = 0)
{
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
  int x  = 0;
  int y  = b;
  int a2 = a * a;
  int b2 = b * b;
  int d = 4 * a2 * b - 4 * b2 - a2;

  drawEllipsePoints(pic, x, y, center.x, center.y, filled, colour);

  while (a2 * y > b2 * x)
    {
      if (d < 0)
        {
```

```
            --y;
            d += 8 * a2 * y - b2 * (8 * x + 12);
          }
        else
          d -= b2 * (8 * x + 12);
        ++x;
        drawEllipsePoints(pic, x, y, center.x, center.y, filled, colour);
      }

  d = 4*b2*a2 - 4*a2*y*y - 4*b2*x*x - 4*b2*x + 4*a2*(2*y-1) - b2;

  while (y >= 0)
    {
      if (d < 0)
        d += a2 * (8 * y - 12);
      else
        {
          ++x;
          d -= 8 * b2 * x - a2 * (8 * y - 12);
        }
      --y;
      drawEllipsePoints(pic, x, y, center.x, center.y, filled, colour);
    }
//>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
}

//########################################################################
int maindraw()
{
  Drawing pic1(200, 200);
```

```cpp
Drawing pic2(pic1);

pic1.show( );
pic1.setZoom(2);
pic2.show( );
pic2.setZoom(2);

IPoint2D center;
int a, b;
int colour;

while (true)
  {
    cout << "Eingabe von center, a, b, colour: ";
    cin >> center >> a >> b >> colour;
    if (center.x < 0 || center.y < 0)
      break;
    pic1.show( );
    drawEllipse(pic1, center, a, b, false, colour);
    cin.get( );
    cout << "Weiter mit Return" << endl;
    cin.get( );
    pic2.show( );
    drawEllipse(pic2, center, a, b, true, colour);
    cout << "Weiter mit Return" << endl;
    cin.get( );
  }

cout << endl;
IOThread::waitForWindow(60);
```

```
    return 0;
}
/*
(100,100) 25 100 0


(100,100) 100 60 70


(10,100) 10 100 100


(-1,-1) 1 1 1
*/
```