Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**Пермский национальный исследовательский**

**политехнический университет**

ЭТФ

ИТАС

Факультет\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Кафедра\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КОБ

Специальность \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ОТЧЕТ**

* **лабораторной работе № 12**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(*наименование лабораторной работы*)

Выполнил:

студент группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КОБ-23-1С

Горкунов М. В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(*фамилия*, *инициалы*)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(*подпись*)

Проверил:

доцент кафедры ИТАС Тарутин А.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(*оценка*)

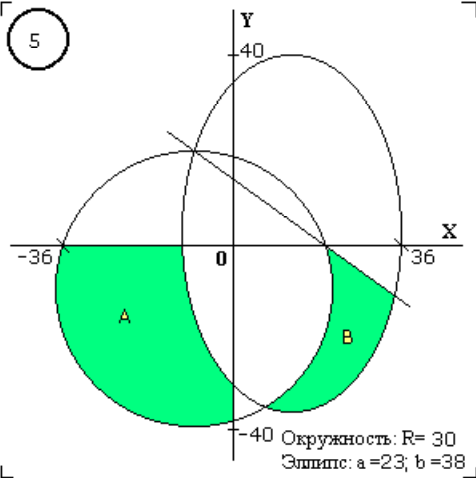
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(*подпись*)

**Цель лабораторной работы:** разработать алгоритм и программу для вычесления координат критических точек.

**Содержание лабораторной работы:** применение метода простого сканирования для вычеслентя точек. Для сравнения результатов вычислений координат каждой точки должны быть найдены аналитически. Точность результатов приближенных вычислений – 0,001. Вывести результатов на экран в виде таблицы.

На рисунке изображены фигуры чи критическии точки нужно найти.

\*Условие: критическими называются точки, образованные пересечением двух или трех геометрических объектов, ни одна из координат которых не равна нулю.\*

Для решения задачи понадобятся следующие **формулы:**

Уравнение окружности:

Уравнение эллипса:

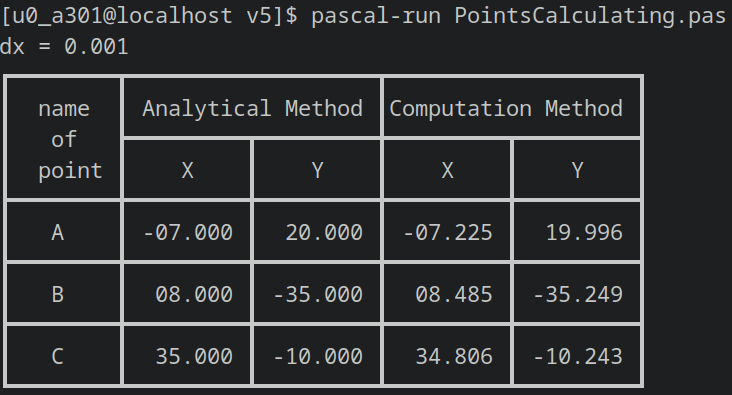
Уравнение прямой:

Алгоритм нахождения координат кретических точекметодом простого сканирования с шагом dx:

1. Определяем ось сканирования ox или oy.
2. Определяем начальную координату сканирования.
3. Определяем напровление сканирование.
4. Определяем условия сканирования.
5. Определяем координаты точки с точностью dx.

После выведем координаты кретических точек с точностью до 0,001 на экран в виде таблицы.

Результат работы:

**Программный код:**

const r = 30.0;

const a = 23.0;

const b = 38.0;

const circleYProjection = -40.0;

const circleXIntersection = -36.0;

const ellipseXIntersectin = 36.0;

const ellipseYProjection = 40.0;

function GetSecondCoordinateOfCircle(coordinate, radius : real) : real; begin

Result := sqrt(sqr(radius) - sqr(coordinate));

end;

function GetSecondCoordinateOfEllipse(coordinate, radius1, radius2 : real) : real; begin

Result := radius1 \* sqrt(1 - sqr(coordinate / radius2));

end;

begin

var dx : real;

if (ParamCount <> 1) or not real.TryParse(ParamStr(1), dx) then

dx := 0.001;

var circleY := circleYProjection + r;

var circleX := circleXIntersection + GetSecondCoordinateOfCircle(-circleY, r);

var ellipseY := ellipseYProjection - b;

var ellipseX := ellipseXIntersectin - GetSecondCoordinateOfEllipse(-ellipseY, a, b);

var circleLineXIntersection := circleX + GetSecondCoordinateOfCircle(-circleY, r);

var circleEllipseIntersectionX := circleXIntersection;

var circleEllipseIntersectionY := circleY + GetSecondCoordinateOfCircle(circleEllipseIntersectionX - circleX, r);

while sqr((circleEllipseIntersectionX - ellipseX) / a) + sqr((circleEllipseIntersectionY - ellipseY) / b) > 1 do begin

circleEllipseIntersectionX += dx;

circleEllipseIntersectionY := circleY + GetSecondCoordinateOfCircle(circleEllipseIntersectionX - circleX, r);

end;

var k := -circleEllipseIntersectionY / (circleLineXIntersection - circleEllipseIntersectionX);

var c := - k \* circleLineXIntersection ;

var circleEllipseIntersectionX2 := circleXIntersection;

var circleEllipseIntersectionY2 := circleY - GetSecondCoordinateOfCircle(circleEllipseIntersectionX2 - circleX, r);

while sqr((circleEllipseIntersectionX2 - ellipseX) / a) + sqr((circleEllipseIntersectionY2 - ellipseY) / b) > 1 do begin

circleEllipseIntersectionX2 += dx;

circleEllipseIntersectionY2 := circleY - GetSecondCoordinateOfCircle(circleEllipseIntersectionX2 - circleX, r);

end;

var ellipseLineIntersectionX := (a \* b \* sqrt(-ellipseY\*\*2 + (2 \* k \* ellipseX + 2 \* c) \* ellipseY - k\*\*2 \* ellipseX\*\*2 - 2 \* c \* k \* ellipseX + a\*\*2 \* k\*\*2 - c\*\*2 + b\*\*2) + a\*\*2 \* k \* ellipseY + b\*\*2 \* ellipseX - a\*\*2 \* c \* k) / (a\*\*2 \* k\*\*2 + b\*\*2);

var ellipseLineIntersectionY := k \* ellipseLineIntersectionX + c;

writelnformat('dx = {0}', dx);

writeln('┏━━━━━━━━┳━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━┓');

writeln('┃ name ┃ Analytical Method ┃Computation Method ┃');

writeln('┃ of ┣━━━━━━━━━┳━━━━━━━━━╋━━━━━━━━━┳━━━━━━━━━┫');

writeln('┃ point ┃ X ┃ Y ┃ X ┃ Y ┃');

writeln('┣━━━━━━━━╋━━━━━━━━━╋━━━━━━━━━╋━━━━━━━━━╋━━━━━━━━━┫');

writelnformat('┃ A ┃ -07.000 ┃ 20.000 ┃ {0: 00.000;-00.000} ┃ {1: 00.000;-00.000} ┃',

circleEllipseIntersectionX, circleEllipseIntersectionY);

writeln('┣━━━━━━━━╋━━━━━━━━━╋━━━━━━━━━╋━━━━━━━━━╋━━━━━━━━━┫');

writelnformat('┃ B ┃ 08.000 ┃ -35.000 ┃ {0: 00.000;-00.000} ┃ {1: 00.000;-00.000} ┃',

circleEllipseIntersectionX2, circleEllipseIntersectionY2);

writeln('┣━━━━━━━━╋━━━━━━━━━╋━━━━━━━━━╋━━━━━━━━━╋━━━━━━━━━┫');

writelnformat('┃ C ┃ 35.000 ┃ -10.000 ┃ {0: 00.000;-00.000} ┃ {1: 00.000;-00.000} ┃',

ellipseLineIntersectionX, ellipseLineIntersectionY);

writeln('┗━━━━━━━━┻━━━━━━━━━┻━━━━━━━━━┻━━━━━━━━━┻━━━━━━━━━┛');

end.

**Вывод:** в данной лабораторной работе был разработан алгоритм и написана программа для вычисления координат критических точек.