МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Вятский государственный университет»
Факультет автоматики и вычислительной техники
Кафедра электронных вычислительных машин

# СОЗДАНИЕ АНИМАЦИИ

Отчет по лабораторной работе №3 дисциплины
«Компьютерная графика»

Выполнил студент группы ИВТ-21 _____/Рзаев А.Э./
Проверил старший преподаватель _____/Вожегов Д.В./

2016 г.

## 1 Постановка задачи

Реализовать демонстрационный ролик (анимацию) движения объектов прямолинейно и по окружности

## 2 Словесное описание алгоритма

Для движения объекта по окружности можно использовать функции синуса и косинуса, применяя их соответственно к координатам X и Y, также необходимо учитывать радиус и центр траектории движения объекта. В результате получаем 2 уравнения для расчёта координат:

$X = X0 + R*Cos(a)$

$Y = Y0 + R*Sin(a)$,

где X0, Y0 – координаты центра траектории, R – радиус траектории, a – фаза движения (в радианах).

Соответственно, для анимации потребуется изменять фазу, пересчитывать координаты объекта и обновлять поле для рисования.

## 3 Вывод

Для приведённого выше описания алгоритма была написана программа, реализующая его. Данная программа является простейшим примером создания анимации. Блок-схема алгоритма перерисовки поля, листинг процедур и экранные формы приведены в приложениях А, Б и В.

Приложение А
(обязательное)
Блок-схемы алгоритмов

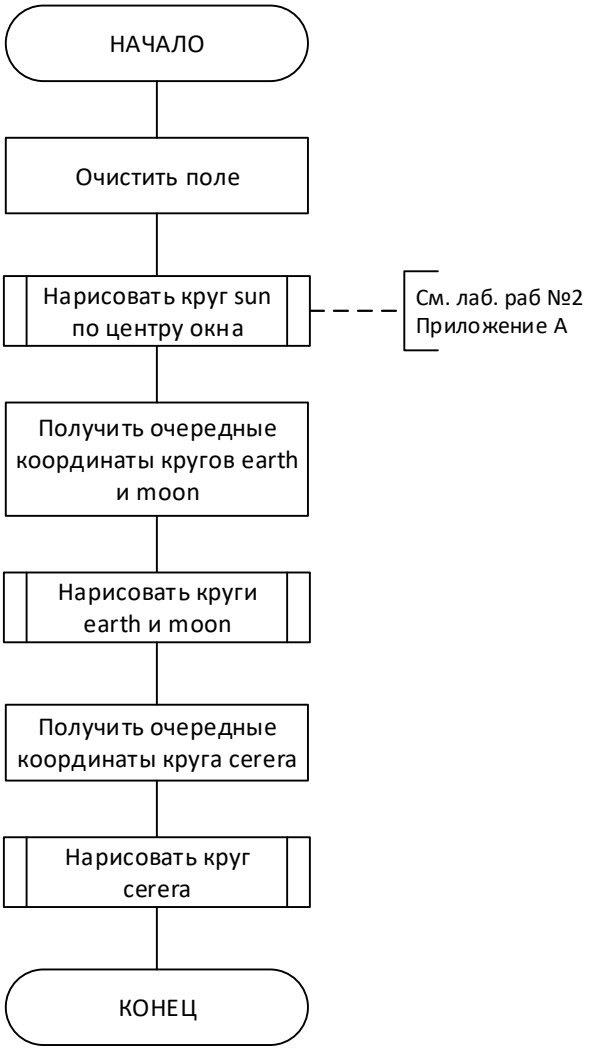


Рисунок А.1 – Схема алгоритма перерисовки изображения

# Приложение Б
(обязательное)
Листинг процедур

```
def orbit(center, step, radius):
    # генератор координат вращающихся кругов
    from math import sin, cos, pi
    angle = 0
    while True:
        x = center()[0] + radius * cos(angle)
        y = center()[1] + radius * sin(angle)
        angle = (angle + step) % (2 * pi)
        yield x, y


def linear(start, step, length):
    # генератор координат прямолинейно движущихся кругов
    pos = 0
    while True:
        ret = pos + start
        pos = (pos + step) % length
        yield ret


def update_image(self):
    def drawer(color):
        return lambda x, y: self.image.setPixel(x, y, color)

    self.image.fill(QtGui.QColor(0, 0, 0))

    x, y = self.earth_pos = self.orbit_earth.__next__()
    self.circle_brezenhem(self.center[0], self.center[1], self.data_sun[0],
drawer(0xffff00))  # sun
    self.circle_brezenhem(x, y, self.data_earth[1], drawer(0x00ff00))  # earth
    x, y = self.orbit_moon.__next__()
    self.circle_brezenhem(x, y, self.data_moon[1], drawer(0xff00ff))  # moon
    y = self.orbit_cerera.__next__()
    self.circle_brezenhem(20, y, self.data_cerera[0], drawer(0xffffff))

    self.image_item.setPixmap(QtGui.QPixmap.fromImage(self.image))
```

Приложение В
(обязательное)
Экранные формы программы

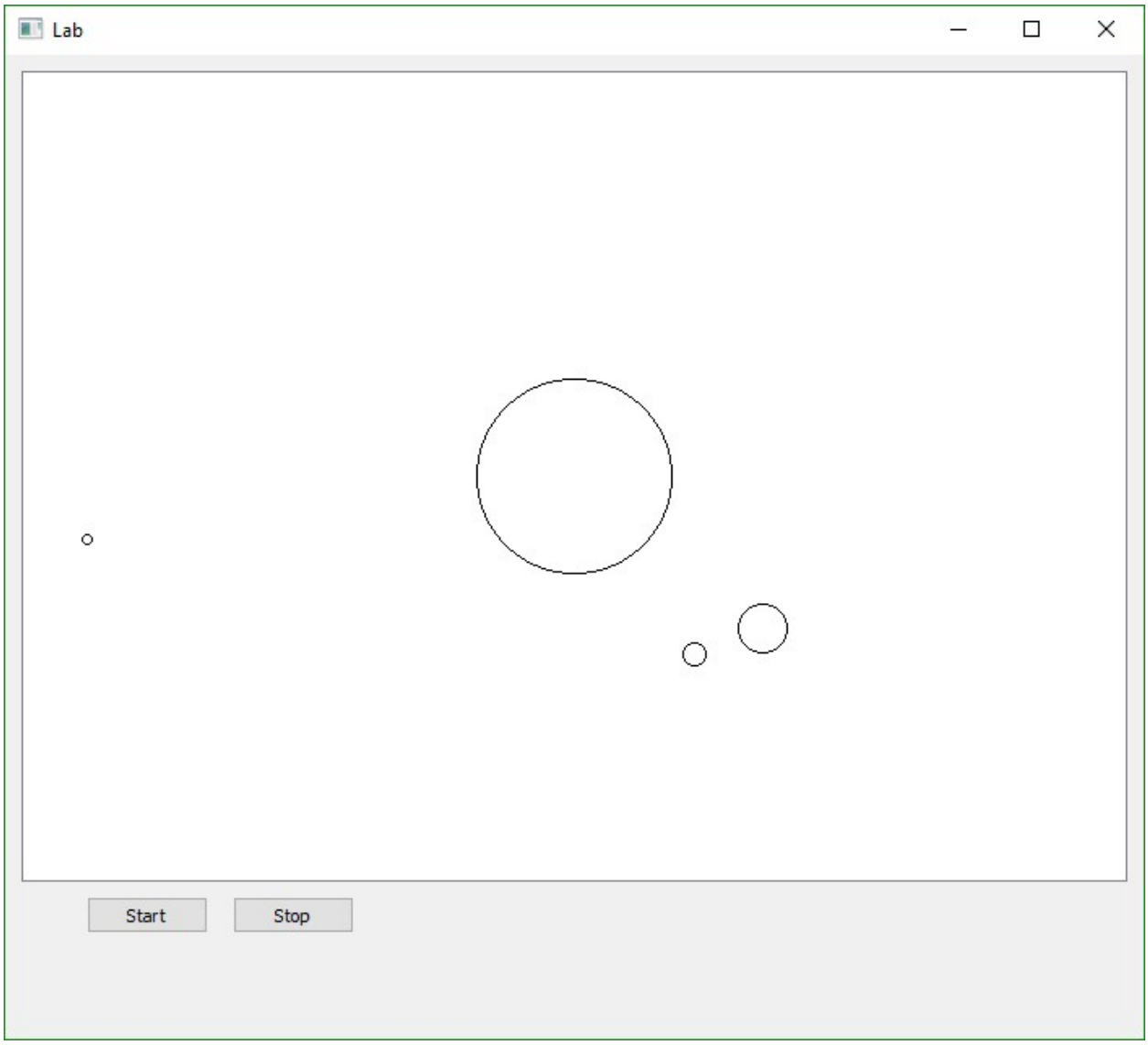


Рисунок В.1 – Основной экран программы