**Лабораторная работа 3 Филон И.Ю. 1 группа**

**Цель:**

Изучить теорию и применить изученный материал на практике по использованию базовых алгоритмов растеризации отрезков и кривых:

1. Пошаговый алгоритм
2. Алгоритм ЦДА
3. Алгоритм Брезенхема
4. Алгоритм Брезенхема(окружность)

**Задачи:**

1. Реализовать вышеперечисленные методы (для одного продемонстрировать вычисления)
2. Интерфейс пользователя должен быть удобен и понятен
3. Отрисовать сетку, а на ней показать работу алгоритмов

**Использованные средства разработки:**

* Java + Java Swing

**Ход работы:**

1. Создание класса Frame для отображения
2. Создание панели scene, которая представляет собой координатную плоскость, для которой задается размер клетки cellSize для изменения масштаба. Чтобы нарисовать сетку и оси реализованы методы drawGrid() и drawAxis(), а для демонстрации работы алгоритмов реализован метод draw(), который в зависимости от выбранного метода вызывает метод отрисовки
3. Создание интерфейса Algorithms с единственным методом draw.
4. Реализация алгоритма ЦДА в виде класса DDAAlgorithm, которой реализует интерфейс Algorithms и переопределяет метод draw()
5. Реализация алгоритма Брезенхема в виде класса BresenhamAlgorithm, которой реализует интерфейс Algorithms и переопределяет метод draw()
6. Реализация алгоритма Брезенхема для окружности в виде класса BresenhamCircleAlgorithm, которой реализует интерфейс Algorithms и переопределяет метод draw()
7. Реализация пошагового алгоритма в виде класса StepByStepAlgorithm, которой реализует интерфейс Algorithms и переопределяет метод draw()
8. Для алгоритма ЦДА добавлено логирование вычислений
9. Для каждого алгоритма добавлено измерение времени выполнения

**Измерение времени работы:**

Тестовые входные данные для отрезка:

x0 = -85 x1 = 75

y0 = -70 y1 = 80

для окружности:

x0 = 0 y0 = 0 r = 75

|  |  |  |
| --- | --- | --- |
|  | Без вывода вычислений (сек) | С вычислениями (сек) |
| ЦДА | 0.016 | 0.013 |
| Пошаговый | 0.019 | - |
| Брезенхема | 0.003 | - |
| Брезенхема для окружности | 0.005 | - |

Видно, что вывод вычислений почти не влияет на время выполнения алгоритмов. Нетрудно заметить, что алгоритм Березенхема как для растеризации отрезка, так и для растеризации окружности работает быстрее своих конкурентов, это связано с тем, что данный алгоритм работает только с целочисленными данными, однако он может терять в точности, если необходимо рисовать наклонные прямые. Напрашивается вывод, что основное время выполнения приходится на отрисовку пикселей, а не на вычисления, а кол-во пикселей одинаковое для всех алгоритмов.

**Вывод:**

В ходе выполнения данной работы я:

* Реализовал приложение, которое позволяет растеризовать отрезки и кривые
* Применил на практике знания, полученные на лекции, закрепил их
* Получил опыт разработки приложений с пользовательским интерфейсом
* Получил опыт работы с Git