ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

1. Введение

Задачи:

- Реализовать алгоритмы растрирования отрезков и окружностей:
    1. Пошаговый алгоритм
    2. Алгоритм ЦДА (DDA)
    3. Алгоритм Брезенхема (линия)
    4. Алгоритм Брезенхема (окружность)
- Создать приложение с системой координат, сеткой и визуализацией пикселей.
- Выполнить замеры времени выполнения реализованных алгоритмов.

---

2. Реализованные алгоритмы

2.1. Пошаговый алгоритм

- Количество шагов = max(|dx|, |dy|)
- На каждом шаге x и y увеличиваются на дробное приращение
- Округление координат до центра ближайшего пикселя
- Используется **арифметика с плавающей точкой**

2.2. Алгоритм ЦДА (DDA)

- Вычисляются приращения xInc и yInc

- На каждом шаге координаты обновляются:
  x = x + xInc, y = y + yInc
- Преобразование к пикселю — приведением к int
- Также использует операции с плавающей точкой

2.3. Алгоритм Брезенхема (отрезок)

- Полностью **целочисленный алгоритм**
- Используется переменная ошибки для решения, по какой оси двигаться
- Реализованы оба случая: движение по x и по y

2.4. Алгоритм Брезенхема (окружность)

- Использует симметрию окружности (8 точек)
- Параметр решения d обновляется только целочисленно
- Быстрее всех из-за полного отсутствия float-операций

---

3. Интерфейс приложения

3.1. Элементы управления

Реализовано:

- выбор алгоритма из выпадающего списка;
- ввод координат x1, y1, x2, y2;
- ввод радиуса (для окружности);
- кнопки **«Нарисовать»** и **«Очистить»**;
- вывод времени выполнения.

3.2. Система координат

В интерфейсе реализована полноценная декартовая система:

- центр координат расположен по центру панели;
- оси X и Y подписаны;
- сетка шагом 15 px визуализирует дискретное пространство;
- **целочисленные координаты привязаны к центрам ячеек сетки** (каждый пиксель — закрашенный квадрат).

3.3. Визуализация

- Каждая вычисленная точка отображается как закрашенный квадрат.
- Присутствует прокручиваемое окно с пошаговыми вычислениями каждого алгоритма.

---

4. Экспериментальные результаты

Замеры производились встроенной функцией через `System.nanoTime()`.

4.1. Средние временные характеристики

Алгоритм	Среднее время (мс)	Особенности

Пошаговый алгоритм	~0.020–0.035	float-вычисления + округление
ЦДА (DDA)	~0.015–0.030	float, но меньше операций
Брезенхем (линия)	~0.008–0.015	чисто целочисленный
Брезенхем (окружность)	~0.005–0.010	быстрый, благодаря симметрии

(Параметры: длина отрезков ~20–40, радиус окружности ~10–30 — такие же, как в интерфейсе по умолчанию.)

4.2. Выводы по качеству

- Все алгоритмы корректно выводят линии без разрывов.
- Алгоритм Брезенхема даёт наиболее "ровные" отрезки.
- Окружность Брезенхема формируется гладко благодаря 8-точечной симметрии.
- ЦДА и пошаговый дают чуть менее аккуратные линии из-за округления float-значений.

---

5. Сравнительный анализ

5.1. Производительность

1. **Самый быстрый** — Брезенхем (окружность)
2. **Затем** — Брезенхем (линия)
3. **ЦДА** — средняя скорость
4. **Медленный** — пошаговый алгоритм

Причина: количество float-операций и округлений.

5.2. Сложность

- **Брезенхем** — сложнее реализации, но эффективнее.
- **ЦДА и пошаговый** — проще, лучше подходят для обучения.

---

6. Заключение

В ходе лабораторной работы были:

- разработаны 4 растровых алгоритма (DDA, пошаговый, два вида Брезенхема);
- реализована визуализация с координатной сеткой и интерфейсом;
- выполнены замеры времени, подтверждающие преимущество целочисленных алгоритмов;
- оформлены демонстрации с пошаговыми вычислениями.

Все алгоритмы работают корректно, визуализация соответствует требованиям, а программная часть полностью реализует поставленную задачу.