|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа № \_3\_**

|  |  |
| --- | --- |
| **Тема \_Рисование отдельных отрезков и сравнение их визуальных характеристик**  **Студент \_Турсунов Ж. Р.\_\_\_\_\_\_\_\_**  **Группа \_ИУ7-46Б\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Куров А. В.\_\_\_\_** |  |

Москва.

2020 г.

**Цель работы:** Рисование отдельных отрезков и сравнение их визуальных характеристик.

**Входные данные:**  Координаты начала и конца для отрезка, цвет отрезка, цвет холста(фона), угол поворота для построения солнца, выбор метода для построения прямой.

**Выходные данные:** Пользовательское меню, содержащее поля ввода и холст с конечным изображением.

**Ошибочные ситуации:** Программа прекращается, если хотя бы один из входных данных не корректен.

**Теоретическая часть**

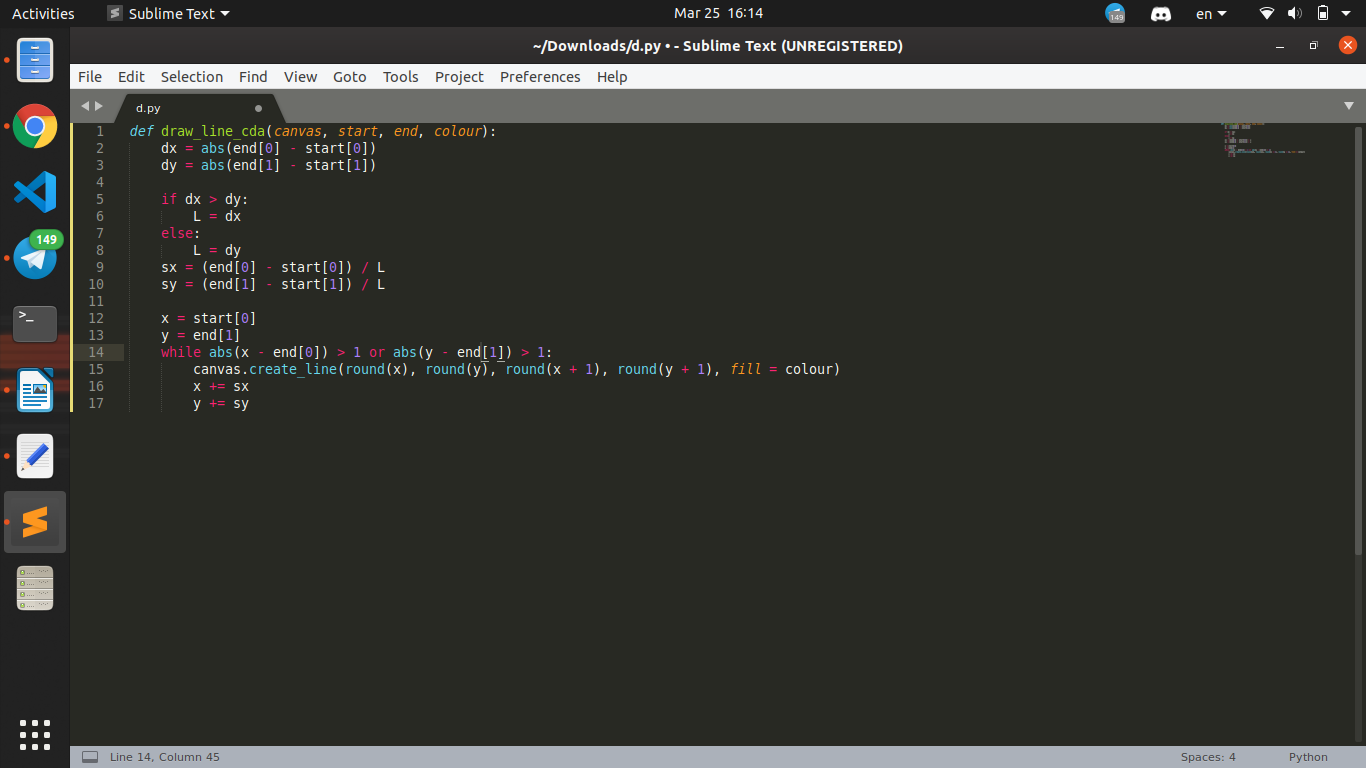
Существуют различные алгоритмы для построения отрезков . Но для всех алгоритмов приписываются одни и те же требования. Прежде всего, генерируемые изображения должны выглядеть как исходные геометрические объекты, то есть отрезки прямых должны выглядеть прямыми Яркость отдельных пикселей, образующих то или иное изображение, должна оставаться постоянной. Рисование должно осуществляться быстро. Собственно все эти требования, не всегда выполняются, особенно первое требования, так как применяя наш алгоритм мы меня аналоговую величину мы меняем на дискретную величину. Естественно, любой алгоритм должен работать быстро. Это требование удовлетворяется путем уменьшения количества вычислений, использованием целочисленной арифметики.

В силу этого нельзя провести отрезок непосредственно из одной точки в другую точку. Поэтому необходимы специальные алгоритмы, позволяющие выбирать пиксели, аппроксимирующие отрезок. Такой процесс определения пикселей, наилучшим образом аппроксимирующих заданный отрезок, называется разложением отрезка в растр.

Алгоритмы построения отрезка:

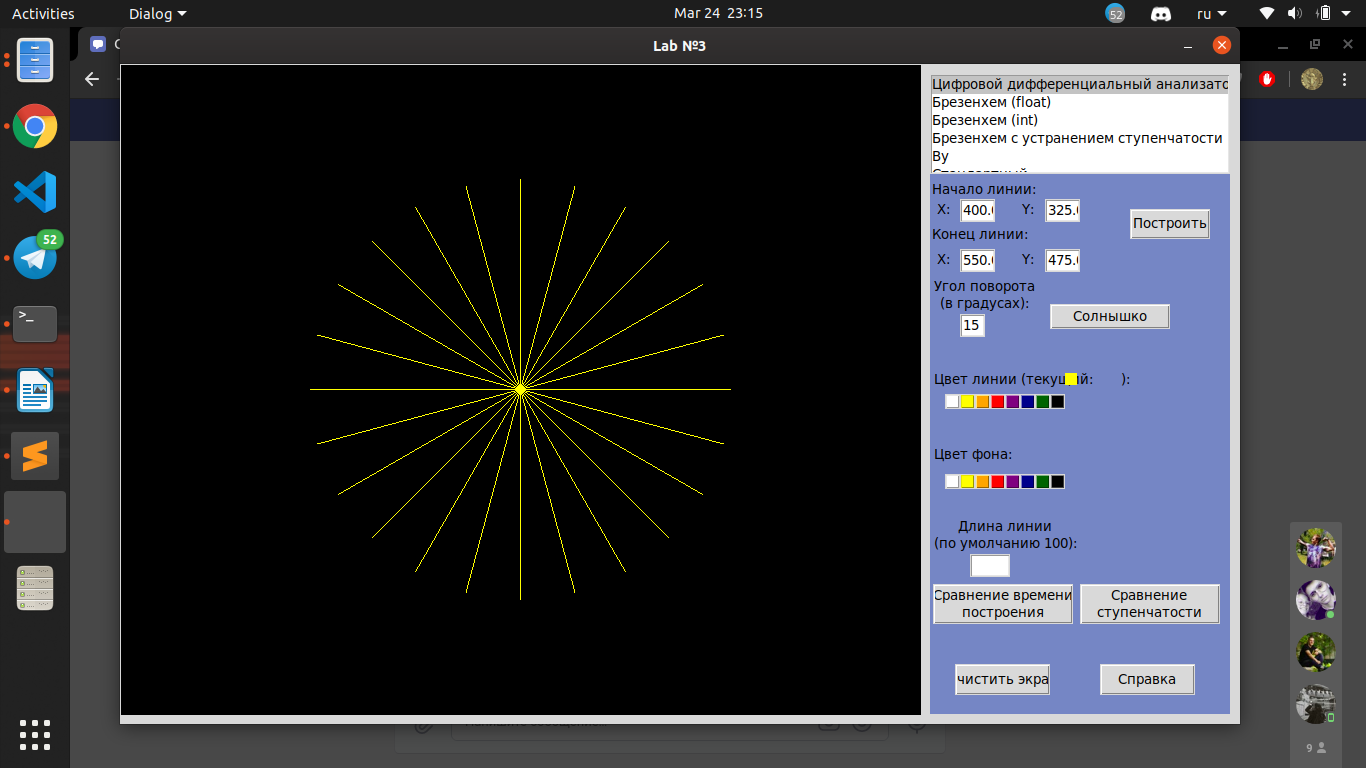
* Алгоритм цифрового дифференциального анализатора.
* Алгоритмы Брезенхема.
* Алгоритм Ву.
* Библиотечный алгоритм.

**Алгоритм цифрового дифференциального анализатора**



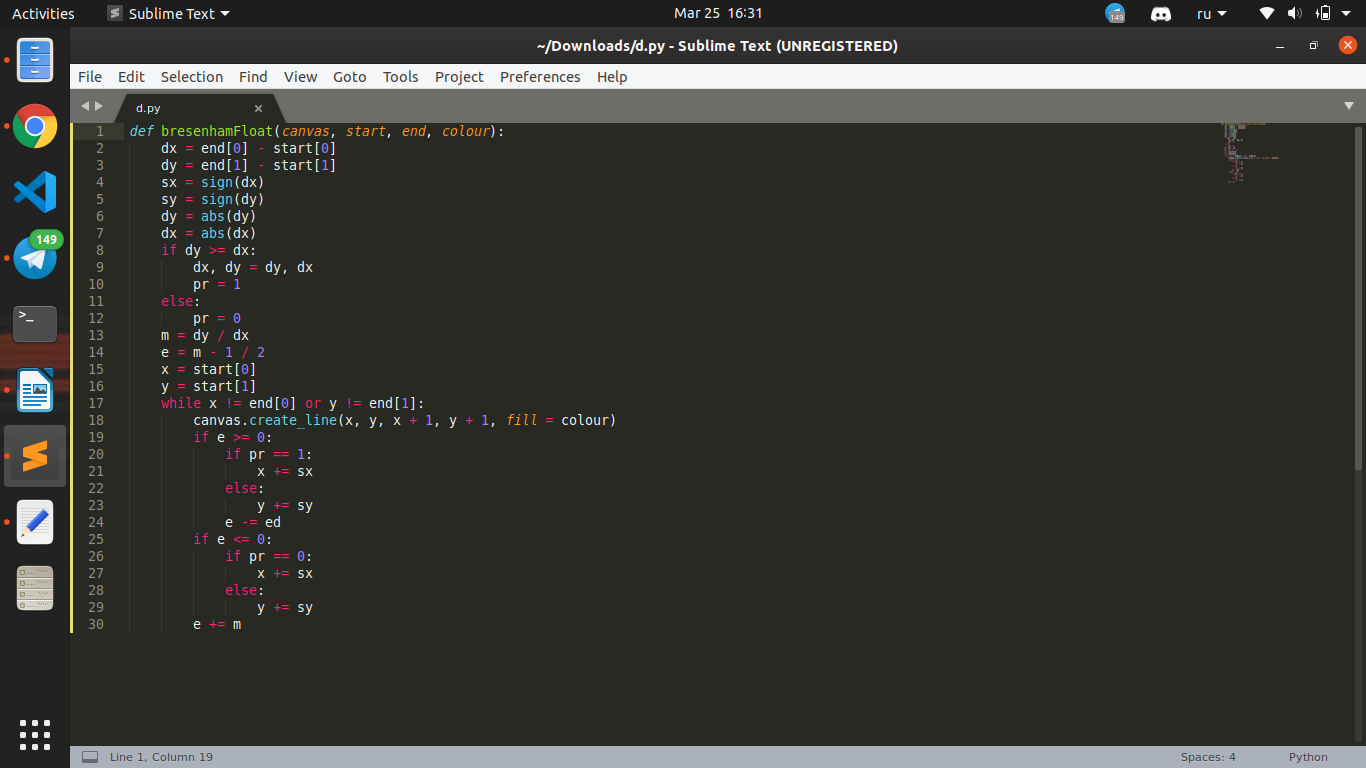
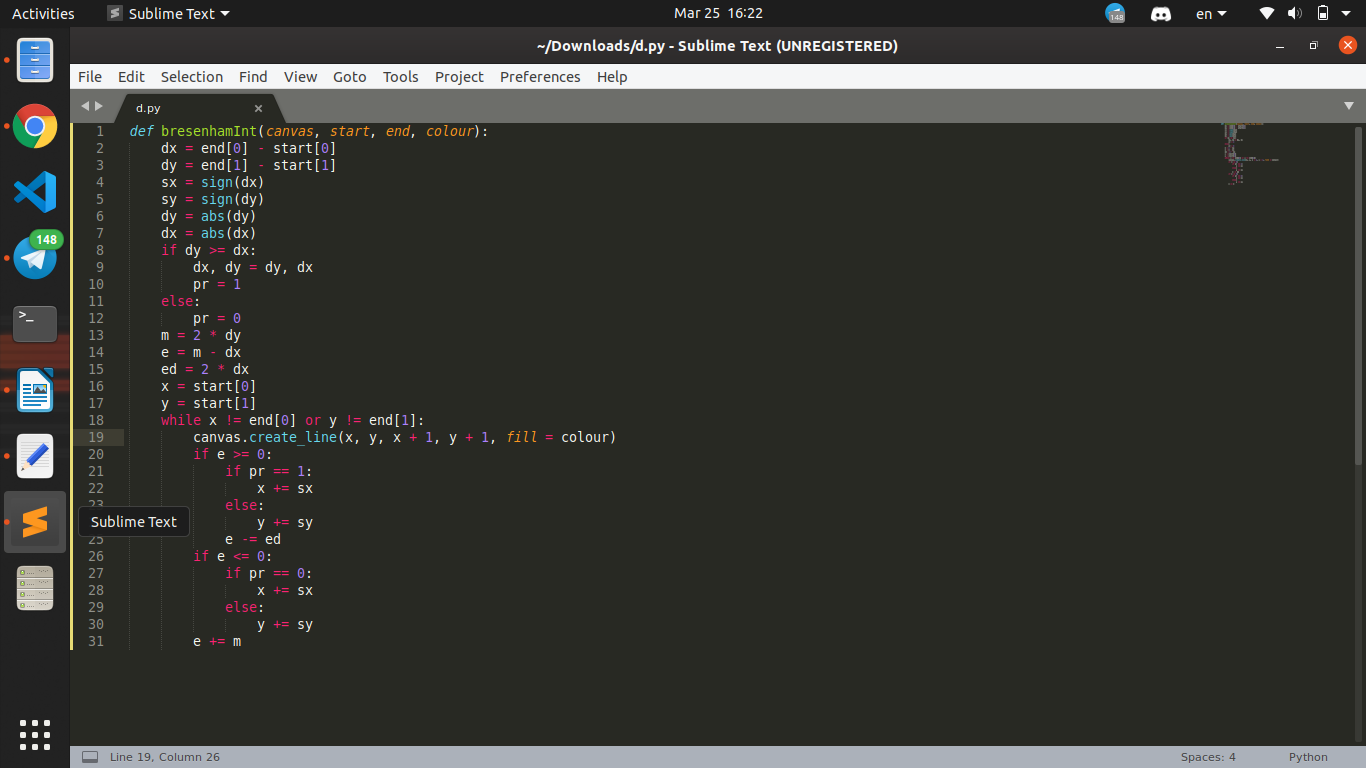
Основным недостатком алгоритма, приведенного выше является работа с числами в формате плавающей точкой. Этого можно избежать, используя арифметику с фиксированной точкой. Несмотря на это, этот алгоритм позволяет понять сложности, которые встречаются при растеризации отрезка.

**Построение солнца**(**угол поворота 15 градусов**)



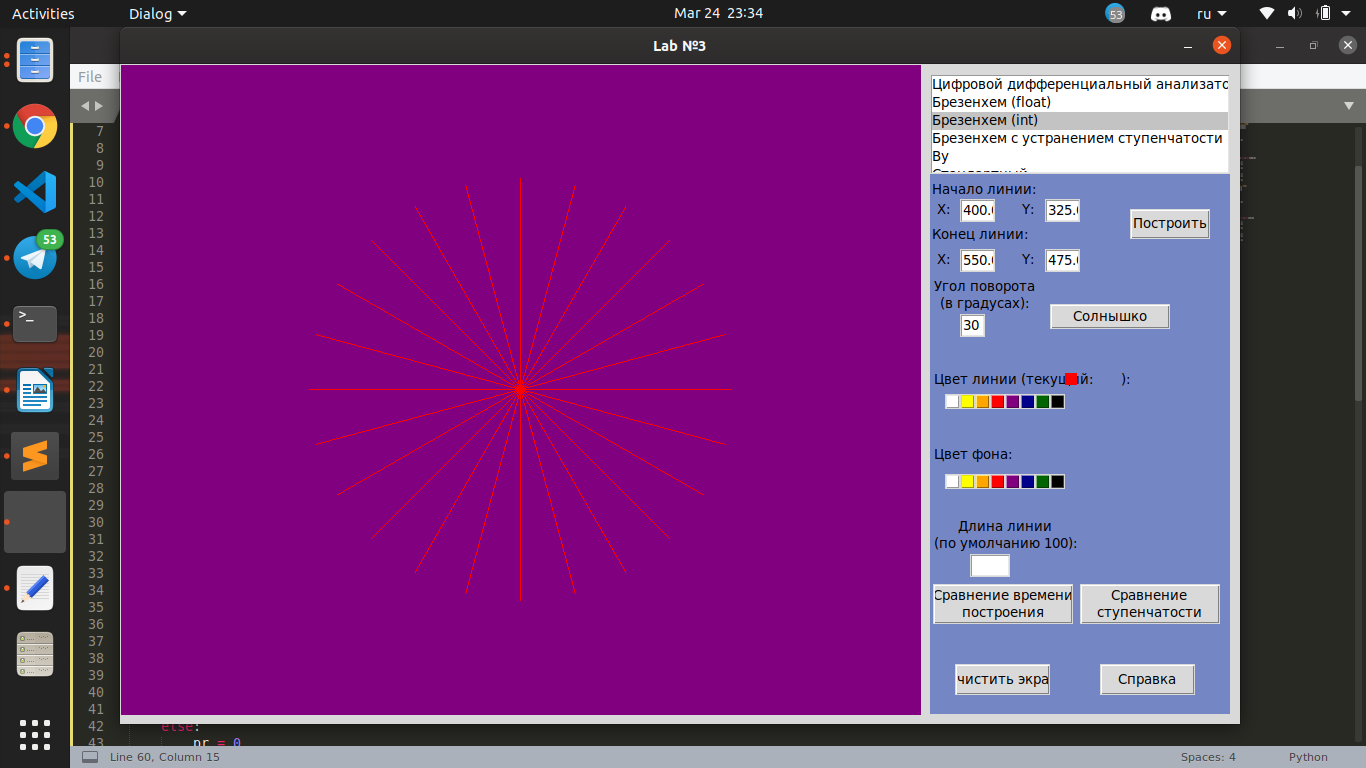
**Алгоритм Брезенхема**

Работа алгоритма Брезенхема основывается на использовании понятия «ошибка». Ошибкой здесь называется расстояние между действительным положением отрезка и ближайшим пикселем сетки растра, который аппроксимирует отрезок на очередном шаге. Недостатком рассмотренного варианта алгоритма Брезенхема является использование вычислений с плавающей точкой и использование операции деления при определении тангенса угла наклона отрезка. Сократить время работы алгоритма можно при использовании целочисленной арифметики и отказа от операции деления.



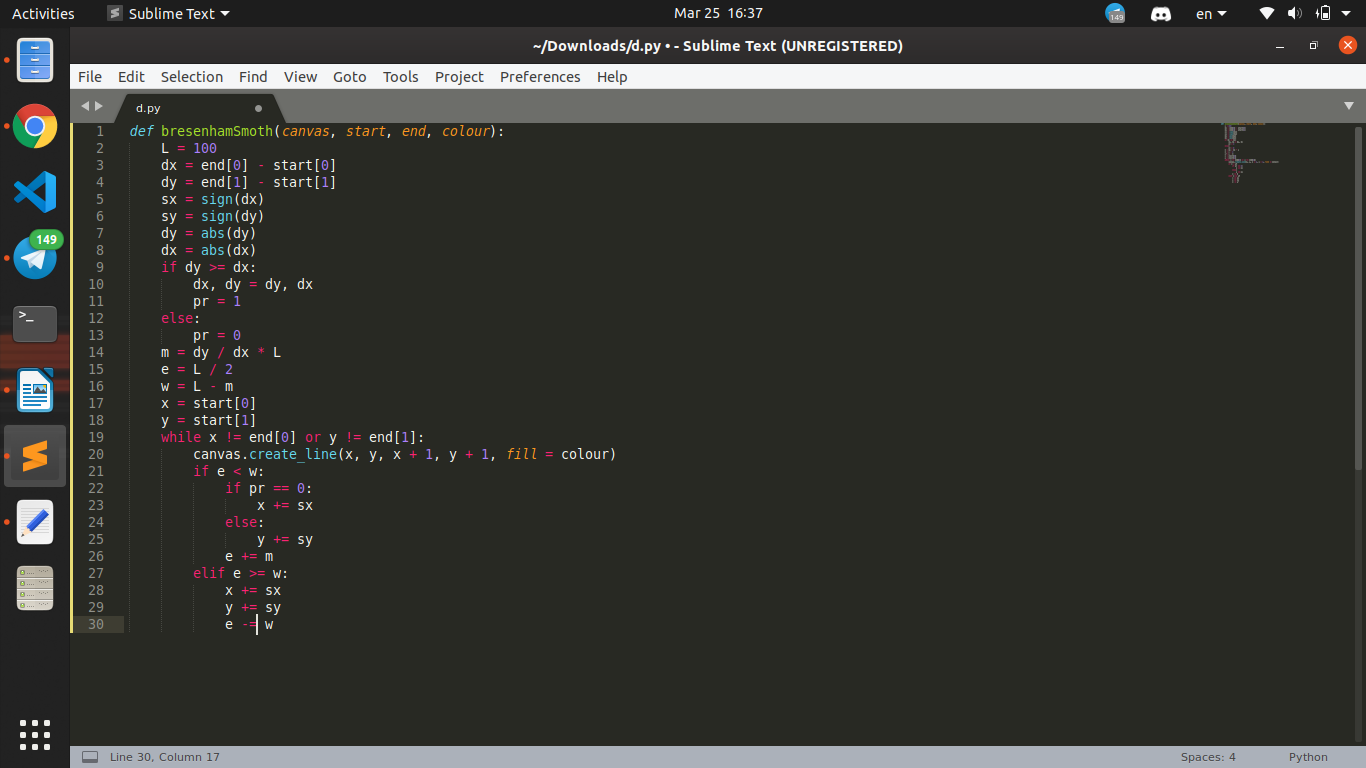
Справа код алгоритма Брезенхема с операцией деления. НО для того чтобы избавиться от этой операции пришлось домножить выражения на 2, что позволяет сократить время. Слева код уже преобразованного алгоритма Брезенхема.

**Построение солнца(угол поворота 15 градусов)**

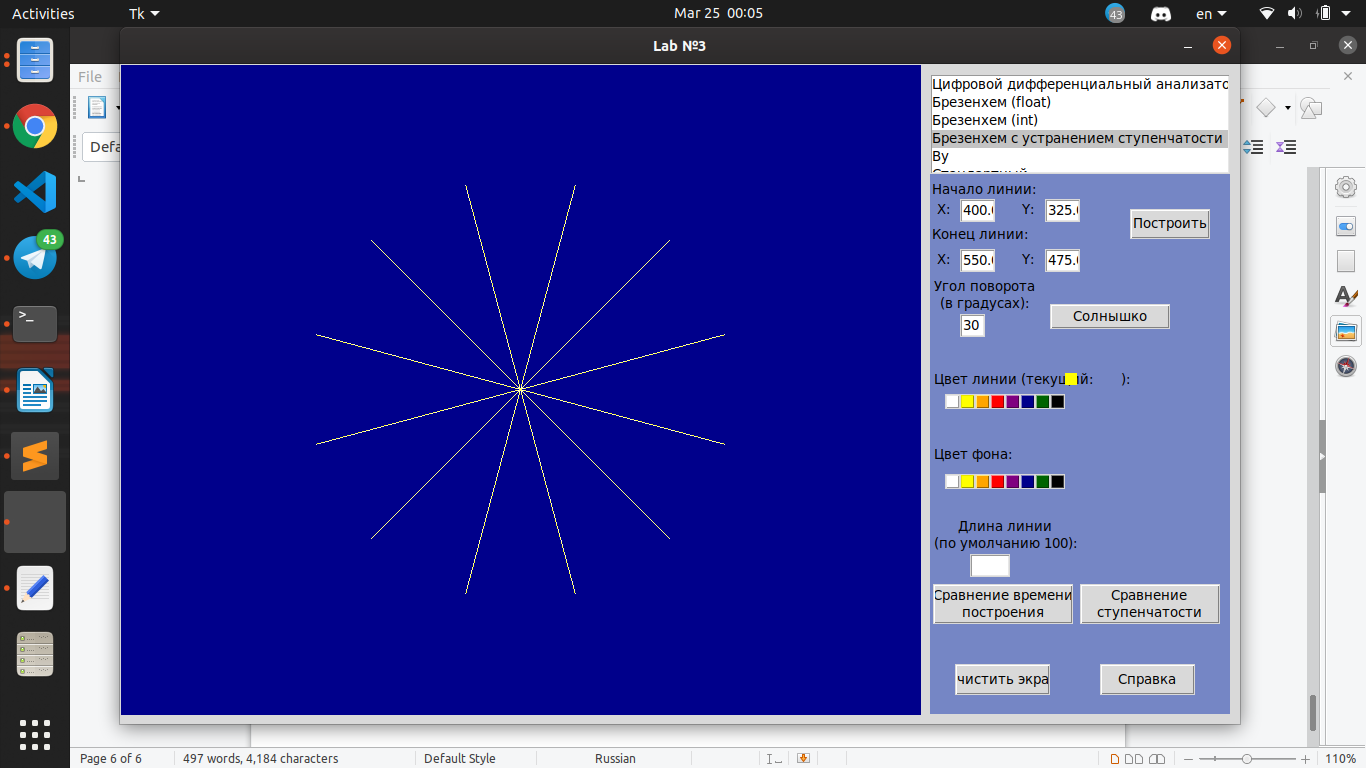


**Алгоритм Брезенхема с устранением ступенчатости**

Ступенчатость — одна из проблем, которую устранить полностью невозможно, но применяя разные методы можно добиться эффекта при котором почти не видны ступеньки. Способ устранения ступенчатости рассматривает пиксель не как математическую точку, а как некоторую конечную область. Для этого интенсивность пикселя устанавливается пропорционально площади части пикселя, находящейся под отрезком. В качестве «ошибки» в данном алгоритме принимается часть площади пикселя, находящаяся под отрезком. Данный алгоритм, как и в предыдущем случае, можно тем же приемом преобразовать к целочисленному виду.

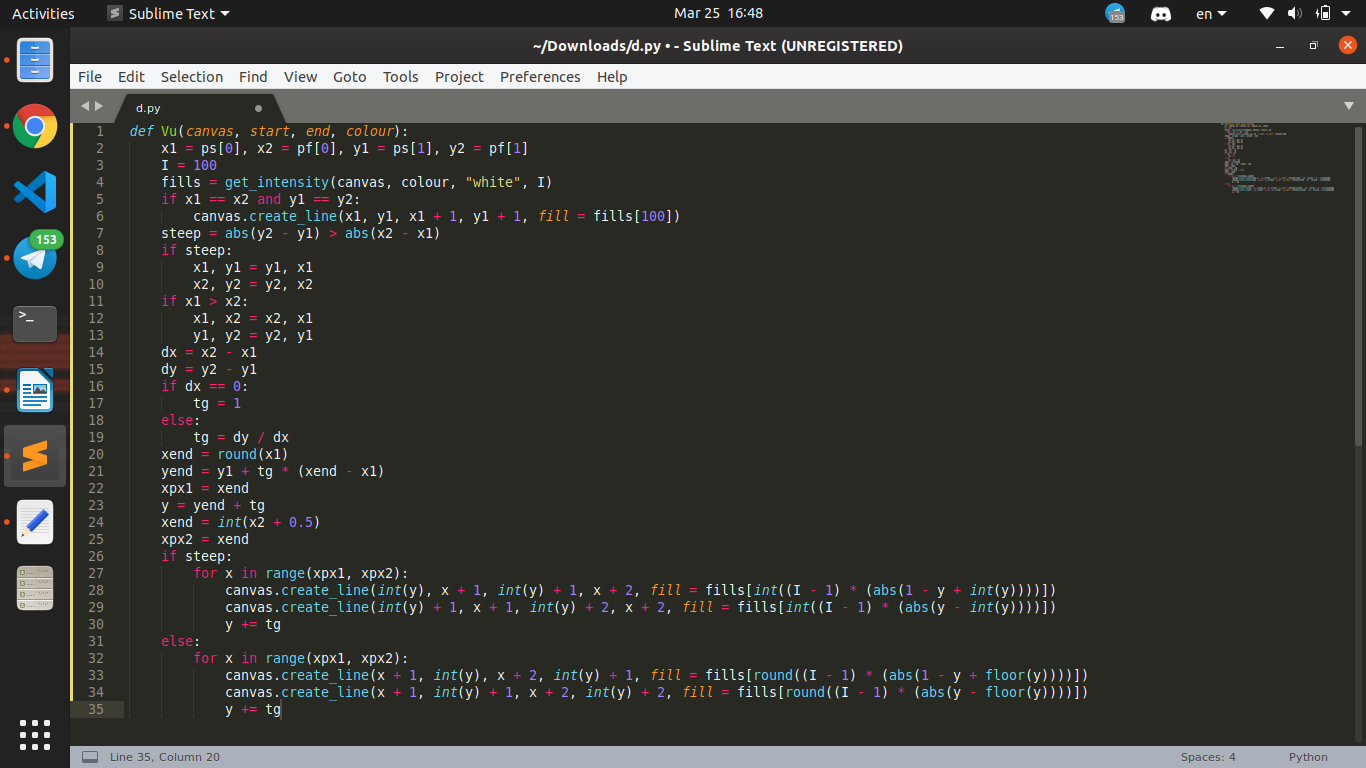


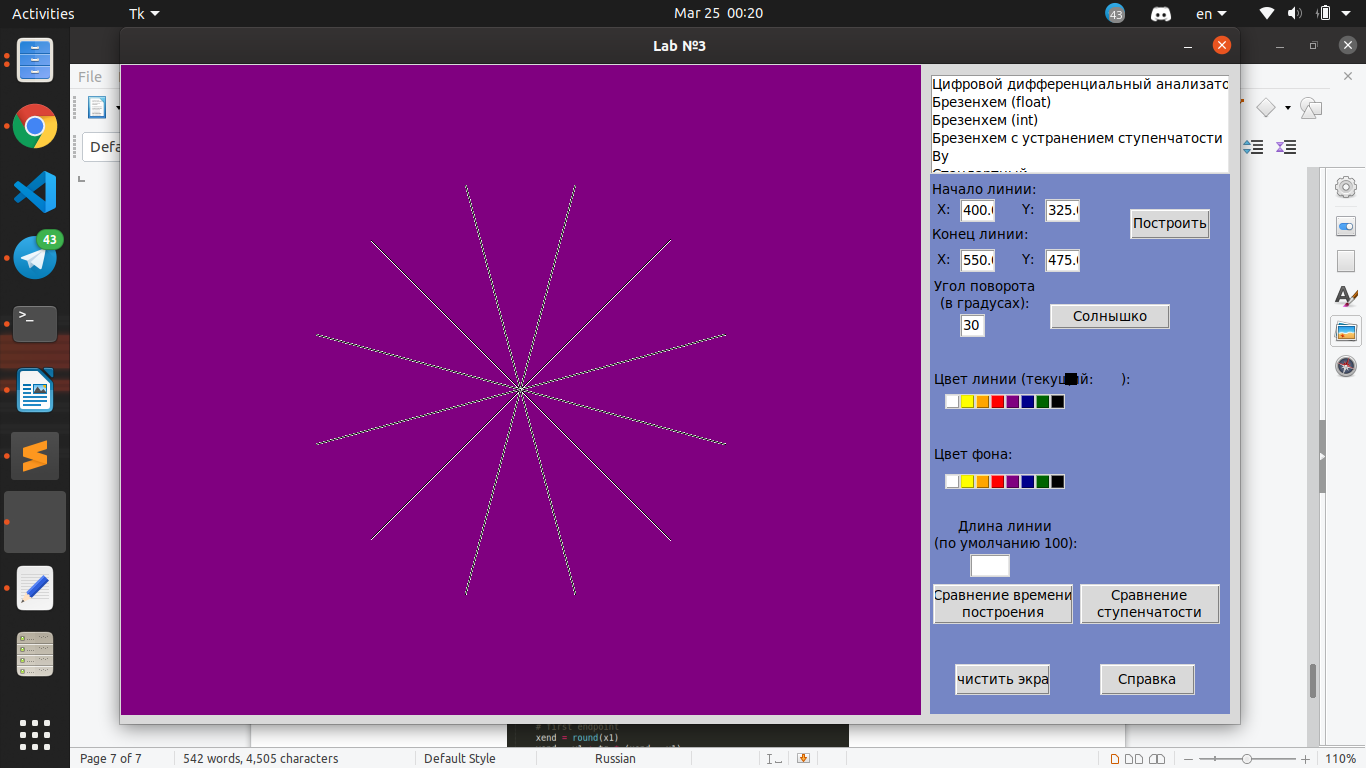
**Построение солнца(угол поворота 30 градусов)**



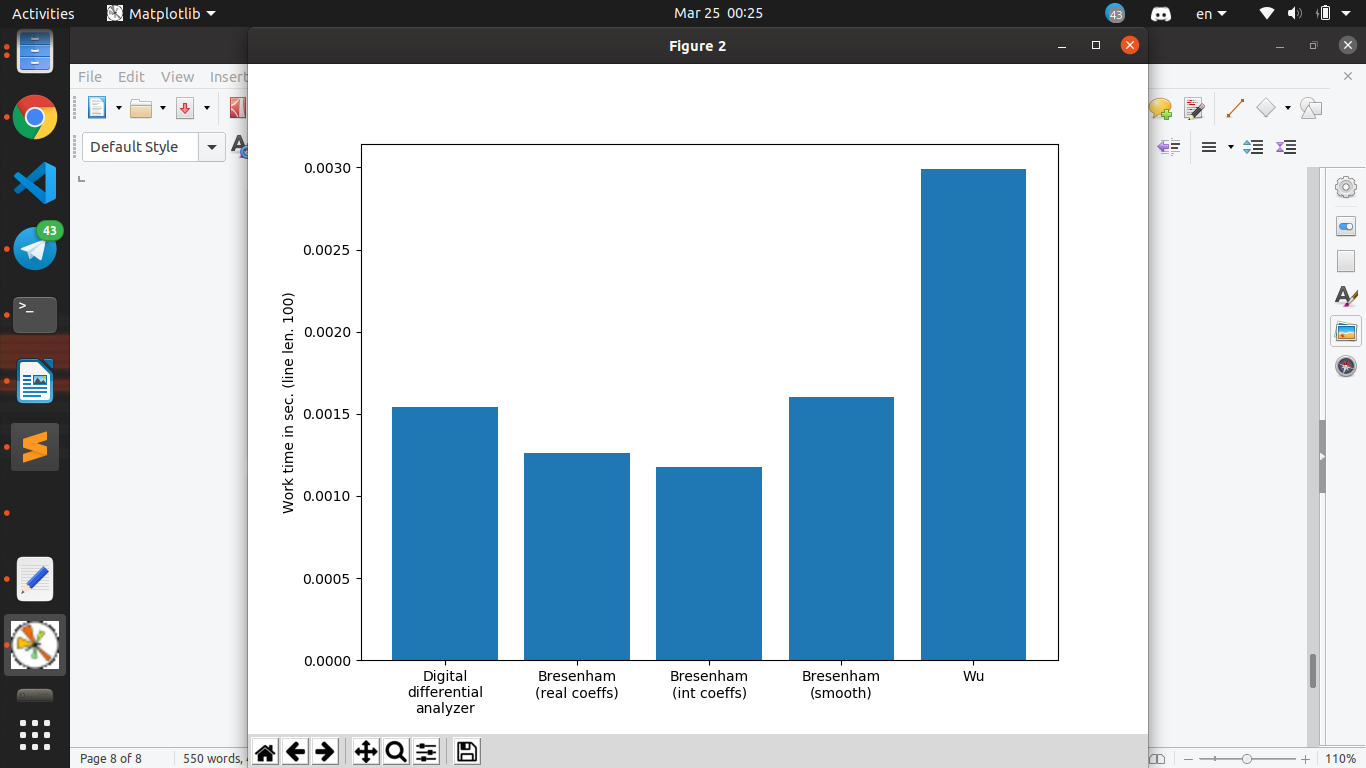
**Алгоритм Ву**

Этот алгоритм отличается от Брезенхема тем, что на каждом шаге ведётся расчёт для двух ближайших к прямой пикселей, и они закрашиваются с разной интенсивностью, в зависимости от удаленности. Иными словами, сто процентов интенсивности делится между пикселями, которые ограничивают векторную линию с двух сторон.



**Построение солнца(угол поворота 30 градусов)**

**Исследование временных характеристик**



**Исследование ступенчатости**

