G01. Алгоритм Брезенхэма для прямой.

Ускорение алгоритма производится за счет рисования только половины отрезка.
Для этого, алгоритм необходимо модифицировать следующим образом:

пусть необходимо нарисовать точку

$$(x_{curr}, y_{curr}) \quad ,$$

тогда необходимо нарисовать и точку

$$(x_2-(x_{curr}-x_1), y_2-(y_{curr}-y_1)) \quad , \text{где}$$

x_1 , y_1 и x_2 , y_2 - начальные и конечные точки по соответствующей координате соответственно.

Данный способ «нахождения» зеркальной точки — теоретический, и, в зависимости от реализации, может быть выражен выражен более удобно, к примеру:

$$(deltaX-1-coordX, deltaY-1-i)$$

(см. код *Bresenham_line.c*).

Результаты замеров показали ускорение как минимум на 12% и, в зависимости от реализации и размера отрезка, может достигать как минимум 20%:

Размер отрезка	Стандартный алгоритм	Ускоренный алгоритм
>1000	0.005195	0.004297
>3000	0.053843	0.044600
>10000	0.351870	0.347963
>30000	3.540673	3.475944

Примеры реализации:

specnano/gcode/Bresenham_line.c
specnano/gcode/Bresenham_line_accelerated.c

G02-G03. Алгоритм Брезенхэма для окружности.

Помимо прямых линий, *алгоритм Брезенхема* может рисовать кривые второго порядка, т.е. существует возможность отрисовки окружности с помощью этого алгоритма. Точки отрисованной кривой, представляющей из себя дугу четверти окружности, зеркально отражаются относительно центра на остальные четверти окружности. Алгоритм требует задания координат центра и радиуса.

Результаты:

Радиус	Время работы
100	0.000740
500	0.021983
1000	0.061979
5000	1.247381
10000	4.601721

Пример реализации:

specnano/gcode/Bresenham_circle.c

Попытка ускорить алгоритм рисованием восьмой и меньших долей окружности приводит к алгоритму *Midpoint Circle Algorithm*.

G02-G03. Midpoint Circle Algorithm.