

Дополнение к отчёту по лабораторной работе №3

1. Пример вычислений для алгоритма Брезенхема (отрезок)

Рассмотрим отрезок от точки A(2, 1) до точки B(10, 6). Для него $dx = 8$, $dy = 5$, $sign(dx) = 1$, $sign(dy) = 1$, а начальное значение ошибки $d_0 = 2 \cdot dy - dx = 2$. Ниже приведены промежуточные значения ошибки и принятое решение (смещение вдоль оси x или одновременно x и y).

i	(x _i , y _i)	d _i	Решение	(x _{i+1} , y _{i+1})	d _{i+1}
0	(2, 1)	2	$d > 0 \Rightarrow (x+1, y+1)$	(3, 2)	-4
1	(3, 2)	-4	$d \leq 0 \Rightarrow (x+1, y)$	(4, 2)	6
2	(4, 2)	6	$d > 0 \Rightarrow (x+1, y+1)$	(5, 3)	0
3	(5, 3)	0	$d \leq 0 \Rightarrow (x+1, y)$	(6, 3)	10
4	(6, 3)	10	$d > 0 \Rightarrow (x+1, y+1)$	(7, 4)	4
5	(7, 4)	4	$d > 0 \Rightarrow (x+1, y+1)$	(8, 5)	-2
6	(8, 5)	-2	$d \leq 0 \Rightarrow (x+1, y)$	(9, 5)	8
7	(9, 5)	8	$d > 0 \Rightarrow (x+1, y+1)$	(10, 6)	2

Итоговый набор целочисленных точек: (2, 1), (3, 2), (4, 2), (5, 3), (6, 3), (7, 4), (8, 5), (9, 5), (10, 6) — совпадает с визуализацией в веб-приложении.

2. Временные характеристики реализованных алгоритмов

Измерения выполнялись в Node.js v18.19.1. Каждый алгоритм вызывался 50 000 раз, после чего среднее время вычислялось как $(\text{общее время}) / 50000$. Для теста линий использован отрезок (2, 1) → (120, 70), для окружности — центр (0, 0) и радиус 60.

Алгоритм	Среднее время за вызов, мс
Пошаговый (отрезок)	0.014
ЦДА (отрезок)	0.015
Брезенхем (отрезок)	0.0010
Брезенхем (окружность)	0.041

Целочисленный Брезенхем для отрезка ожидаемо самый быстрый: он обходится без операций с плавающей точкой и использует минимальный набор вычислений. Алгоритм окружности требует больше времени, потому что на каждом шаге генерирует восемь симметричных точек и обновляет три целочисленных переменных.