LAPORAN GRAFIKA KOMPUTER

“GARIS DDA DAN BRESENHAM PADA GARIS LURUS”

Disusun guna memenuhi tugas mata kuliah Grafika Komputer Dosen Pengampu

Bapak Febi Eka Febriansyah, M.T.,

Wartariyus S.Kom,M.T.I.,

Putut Aji Nalendro, M.PD.

Disusun Oleh:

Nama: Lulu Saputri

NPM: 2413025017

Kelas: PTI 24A

PROGRAM STUDI PENDIDIKAN TEKNOLOGI INFORMASI

JURUSAN PENDIDIKAN MATEMATIKA DAN ILMU PENGETAHUAN ALAM FAKULTAS KEGURUAN DAN ILMU PENDIDIKAN UNIVERSITAS LAMPUNG

2025

Algoritma DDA

DDA adalah metode berbasis perhitungan **numerik incremental** untuk menggambar garis antara dua titik dalam koordinat kartesian. Algoritma ini bekerja dengan menghitung **selisih koordinat (Δx dan Δy)** lalu membagi perbedaan tersebut menjadi langkah-langkah kecil yang menghasilkan garis yang lebih mendekati bentuk idealnya.

Tabel Algoritma DDA

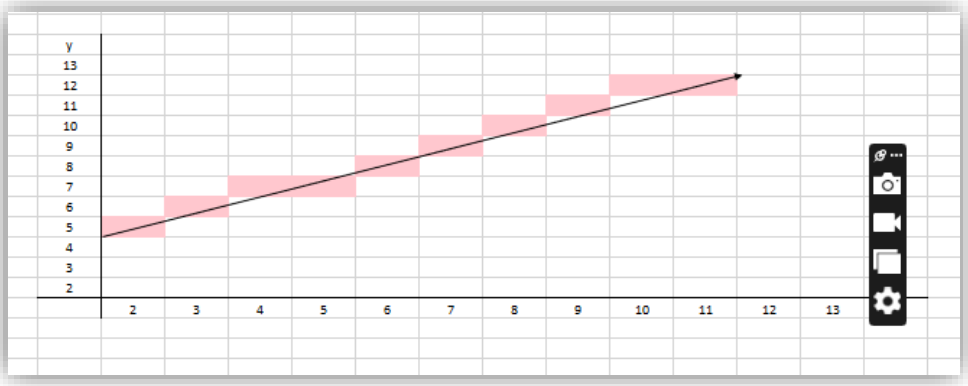
Lulu Saputri - 2413025017

	i		X	Y	(X,Y)
	0		2	5	2,5
	1		3	5,83	3,6
	2		4	6,66	4,7
	3		5	7,49	5,7
	4		6	8,32	6,8
	5		7	9,15	7,9
	6		8	9,98	8,1
	7		9	10,81	9,11
	8		10	11,64	10,12
	9		11	12,47	11,12
	steps		.....		

X0	=	2
Y0	=	5
X1	=	8
Y1	=	10
dx	=	6
dy	=	5
steps	=	6
Xinc	=	1
Yinc	=	0,83

step = jika dx >dy =dx		x = x + xinc	dx = x1 -x0
step = jika dy >dx= dy		y = y + yinc	dy = y1 - y0

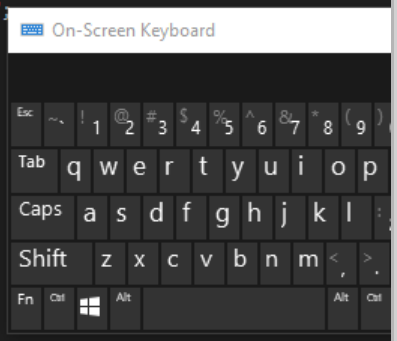
Diagram Algoritma DDA



Program Kodingan Algoritma DDA

C: > Users > lulusaputrii > Documents > semester 2 > grafika komputer > garis DDA.html > html >

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial
scale=1.0">
    <title>Tugas Grafika Komputer Algoritma Garis DDA_Lulu</title>
    <style>
        body{
            background-color : rgb(214, 117, 185);
        }
    </style>
</head>
<body>
    <canvas id="canvasku" width="500" height="500"></canvas>

    <script>
        let canvas = document.getElementById("canvasku");
        let ctx = canvas.getContext("2d");

        ctx.fillStyle = "#99CCFF";
        ctx.fillRect(0,0,500,500);

        function drawDDA(x0,y0,x1,y1){
            let dx = x1 - x0;
            let dy = y1 - y0;

            let step = Math.max(Math.abs(dx), Math.abs(dy));

            let xIncrement = dx / step;
            let yIncrement = dy / step;

            let x = x0;
            let y = y0;

            for (let i = 0; i <= step; i++){
                ctx.fillStyle = "#FF9999";
                ctx.fillRect(x, y, 4, 4);

                x += xIncrement
                y += yIncrement
            }
        }

        // Example: Draw a line from (10,
        drawDDA(450, 50, 50, 450);
    </script>
</body>
</html>
```



Penjelasannya

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial
scale=1.0">
    <title>Tugas Grafika Komputer Algoritma Garis DDA_Lulu</title>
```

<!DOCTYPE html> → Menandakan bahwa ini adalah dokumen HTML5.

<html lang="en"> → Bahasa halaman ini diatur ke bahasa Inggris.

<meta charset="UTF-8"> → Memungkinkan penggunaan berbagai karakter (seperti simbol dan huruf internasional).

<meta name="viewport" content="width=device-width, initial-scale=1.0"> → Membantu tampilan responsif pada perangkat yang berbeda.

<title> → Menentukan judul halaman yang ditampilkan di tab browser.

```
    <style>
        body{
            background-color : rgb(214, 117, 185);
        }
    </style>
```

Memberikan **warna latar belakang pink muda** untuk halaman menggunakan kode RGB **(214, 117, 185)**.

```
    <canvas id="canvasku" width="500" height="500"></canvas>
```

Canvas adalah area gambar dengan ukuran **500x500 piksel**, tempat kita akan menggambar garis.

```
let canvas = document.getElementById("canvasku");
let ctx = canvas.getContext("2d");
```

Mengambil elemen **canvas** dan menyimpannya dalam variabel canvas.

Menggunakan **getContext("2d")** untuk mendapatkan konteks gambar **2D**.

```
21          ctx.fillStyle = "#99CCFF";
22          ctx.fillRect(0,0,500,500);
```

ctx.fillStyle = "#99CCFF"; → Mengatur warna **biru muda** sebagai latar belakang canvas.

ctx.fillRect(0, 0, 500, 500); → Menggambar persegi panjang **500x500** untuk mengisi seluruh area canvas.

```
24      function drawDDA(x0,y0,x1,y1){
25          let dx = x1 - x0;
26          let dy = y1 - y0;
27
28          let step = Math.max(Math.abs(dx), Math.abs(dy));
29
30          let xIncrement = dx / step;
31          let yIncrement = dy / step;
32
33          let x = x0;
34          let y = y0;
35
36          for (let i = 0; i <= step; i++){
37              ctx.fillStyle = "#FF9999";
38              ctx.fillRect(x, y, 4, 4);
39
40              x += xIncrement
41              y += yIncrement
42          }
43      }
```

Cara kerja fungsi drawDDA()

1. **Menghitung perbedaan posisi koordinat**

   dx adalah perubahan nilai **x** antara titik awal dan akhir.

   dy adalah perubahan nilai **y** antara titik awal dan akhir.

2. **Menentukan jumlah langkah (step)**

   Math.abs(dx) dan **Math.abs(dy)** memastikan nilai selalu positif.

   Gunakan nilai terbesar sebagai jumlah langkah (step) untuk memastikan piksel tersebar merata.

3. **Menghitung kenaikan x dan y dalam setiap langkah**

xIncrement adalah jumlah kenaikan koordinat x setiap langkah.

yIncrement adalah jumlah kenaikan koordinat y setiap langkah.

4. **Menggambar titik-titik garis**

   Mulai dari titik awal (x0, y0).

   Menggambar titik persegi kecil **4x4 piksel** sebagai bagian dari garis.

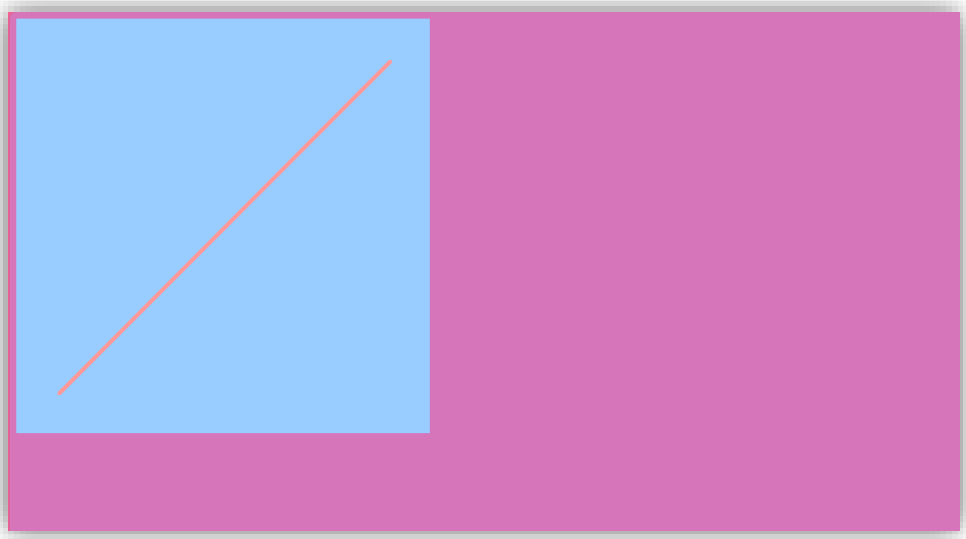
   Setiap iterasi, nilai x dan y bertambah sesuai kenaikan yang dihitung sebelumnya.

```
46          drawDDA(450, 50, 50, 450);
```

Gambar garis dari (450,50) ke (50,450).

Ini berarti garis akan digambar dari **kanan atas ke kiri bawah**.

OUTPUTNYA



Algoritma Bresenham

Bresenham adalah algoritma yang lebih efisien untuk menggambar garis dibandingkan DDA karena **hanya menggunakan operasi integer** (tanpa operasi pembagian dan pembulatan floating-point). Algoritma ini dikembangkan oleh Jack Bresenham pada tahun 1962 dan digunakan secara luas dalam bidang grafika komputer.

Tabel Bresenham

Lulu Saputri - 2413025017

	P		X	Y
	4		2	5
	2		3	6
	0		4	7
	-2		5	8
	8		6	8
	6		7	9
	4		8	10
	2		9	11
	0		10	12
	-2		11	12
	steps			

X0	=	2
Y0	=	5
X1	=	8
Y1	=	10
dx	=	6
dy	=	5
d1	=	10
d2	=	2
p	=	4
m	=	0,83

m=y1-y0/x1-x0

garis0<m<1

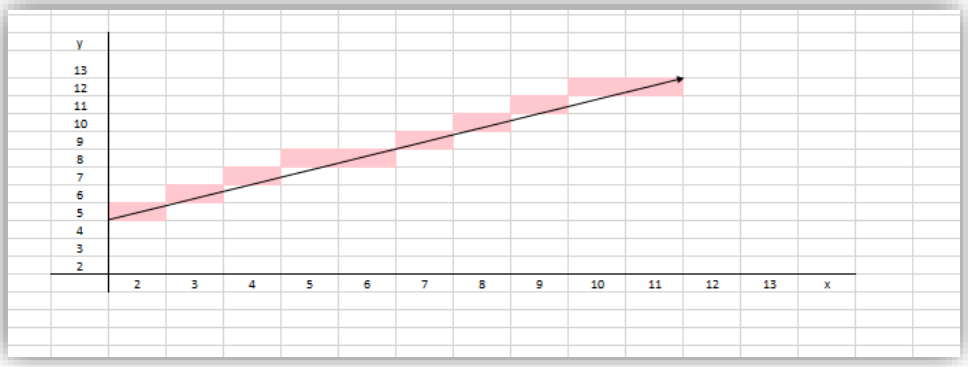
P = d1-dx
jika p >=0 maka:
p= p-d2
y=y+1

jika p <0 maka:
p=p+d1
y=y

d1=2*dy
d2 = 2*(dx-dy)

dx=x1-x0
dy=y1-y0

Diagram Bresenham



Perbedaan Algoritma DDA dan Bresenham

1. Algoritma DDA (Digital Differential Analyzer)

DDA menggambar garis dengan cara menambah nilai x dan y secara bertahap berdasarkan kemiringan garis. Karena ada pembulatan angka desimal, garis yang dihasilkan bisa kurang akurat dan sedikit bergerigi.

2. Algoritma Bresenham

Bresenham menggunakan perhitungan berbasis bilangan bulat, jadi lebih cepat dan lebih akurat dibanding DDA. Algoritma ini memilih piksel mana yang paling dekat dengan garis ideal tanpa perlu operasi pembagian atau pembulatan.