

**LAPORAN PRAKTIKUM**  
**GRAFIKA KOMPUTER**  
(Dosen : *Rio Priantama S.T., M.T.I.*)

**Modul 3**



**Nama : Muhammad Rizal Nurfirdaus**

**NIM : 20230810088**

**Kelas : TINFC-2023-04**

**TEKNIK INFORMATIKA**  
**FAKULTAS ILMU KOMPUTER**  
**UNIVERSITAS KUNINGAN**

## Praktikum

### 1. Praktikum 1 : Implementasi algoritma DDA

Source Code:

```
import matplotlib.pyplot as plt
import numpy as np

# Fungsi untuk menggambar garis menggunakan algoritma DDA
def dda_algorithm(x0, y0, x1, y1):
    # Menghitung perbedaan koordinat
    dx = x1 - x0
    dy = y1 - y0

    # Menentukan jumlah langkah berdasarkan jarak terpanjang (step)
    steps = max(abs(dx), abs(dy))

    # Menghitung perubahan x dan y per langkah
    Xinc = dx / float(steps)
    Yinc = dy / float(steps)

    # Inisialisasi titik awal
    x = x0
    y = y0

    # Menyimpan titik-titik hasil algoritma DDA
    x_points = [x]
    y_points = [y]

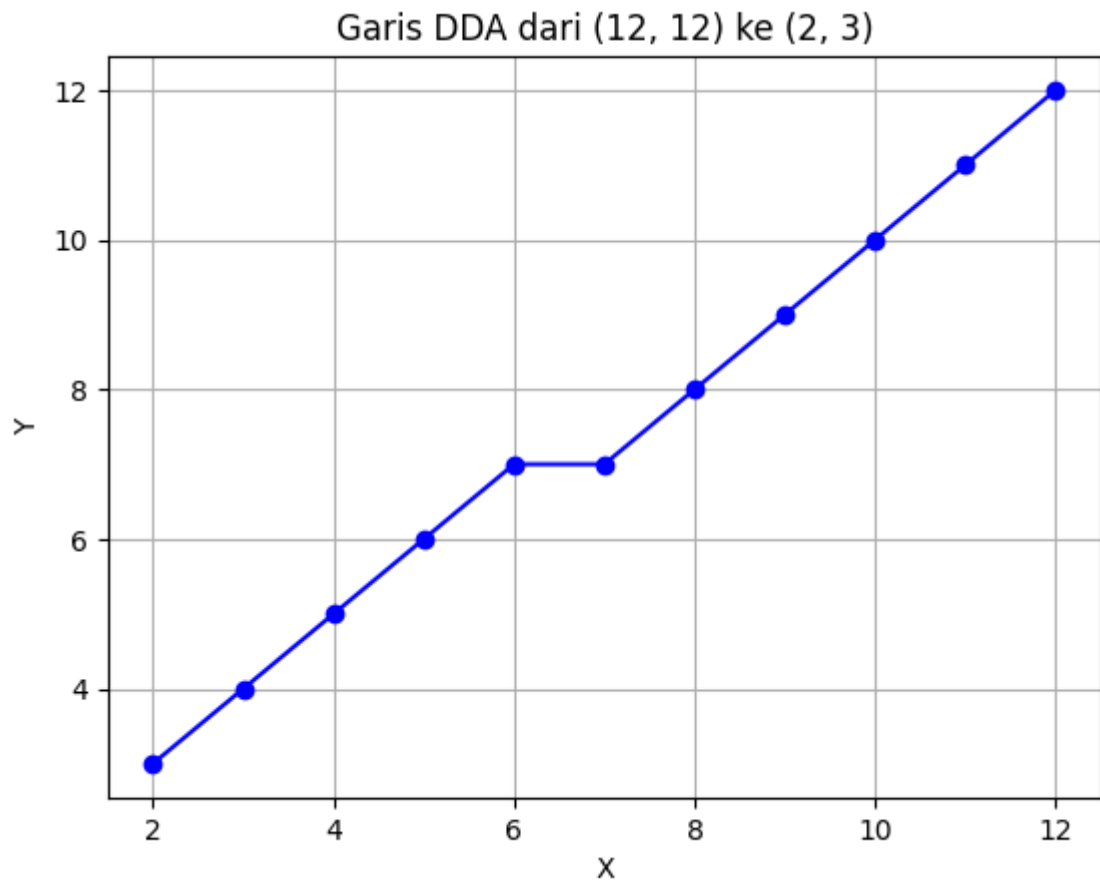
    # Loop untuk menghitung titik-titik sepanjang garis
    for i in range(int(steps)):
        x += Xinc
        y += Yinc
        x_points.append(round(x))
        y_points.append(round(y))

    # Menggambar garis menggunakan matplotlib
    plt.plot(x_points, y_points, marker="o", color="b")
    plt.title(f"Garis DDA dari ({x0}, {y0}) ke ({x1}, {y1})")
    plt.xlabel("X")
    plt.ylabel("Y")
    plt.grid(True)
    plt.show()

# Meminta input dari pengguna untuk titik awal dan titik akhir
x0 = int(input("Masukkan koordinat x0 (titik awal): "))
y0 = int(input("Masukkan koordinat y0 (titik awal): "))
x1 = int(input("Masukkan koordinat x1 (titik akhir): "))
y1 = int(input("Masukkan koordinat y1 (titik akhir): "))

# Memanggil fungsi DDA untuk menggambar garis
dda_algorithm(x0, y0, x1, y1)
```

Hasil Run:



Analisis: Program di atas merupakan implementasi algoritma **DDA (Digital Differential Analyzer)** untuk menggambar garis antara dua titik pada bidang koordinat menggunakan bahasa Python dan library **matplotlib**. Algoritma ini bekerja dengan menghitung perbedaan koordinat antara titik awal ( $x_0, y_0$ ) dan titik akhir ( $x_1, y_1$ ) melalui variabel  $dx$  dan  $dy$ . Kemudian, jumlah langkah (steps) ditentukan berdasarkan nilai mutlak terbesar antara  $dx$  dan  $dy$  agar pergerakan garis tetap proporsional. Nilai perubahan tiap langkah ( $Xinc$  dan  $Yinc$ ) dihitung dengan membagi  $dx$  dan  $dy$  dengan steps, lalu titik-titik koordinat baru diperoleh secara bertahap melalui proses iterasi. Setiap hasil pembulatan dari koordinat tersebut disimpan dalam daftar  $x\_points$  dan  $y\_points$ , yang kemudian divisualisasikan sebagai garis menggunakan fungsi `plt.plot()`. Grafik yang dihasilkan menampilkan titik-titik koordinat hasil perhitungan DDA dalam bentuk garis biru dengan marker bulat, serta dilengkapi dengan label sumbu dan grid untuk memperjelas tampilan. Program ini juga bersifat interaktif karena meminta pengguna memasukkan nilai titik awal dan titik akhir sebelum menggambar garis.

## 2. Praktikum 2 : Implementasi algoritma DDA dengan Input Dinamis

Source Code:

```
import matplotlib.pyplot as plt

# Fungsi untuk menggambar garis menggunakan algoritma Bresenham
def bresenham_algorithm(x0, y0, x1, y1):
```

```

# Inisialisasi variabel
x, y = x0, y0
dx = abs(x1 - x0)
dy = abs(y1 - y0)

# penambahan untuk x dan y (arah)
sx = 1 if x0 < x1 else -1
sy = 1 if y0 < y1 else -1

# Inisialisasi parameter kesalahan (error)
err = dx - dy

x_points = [x]
y_points = [y]

# Loop utama
while x != x1 or y != y1:
    e2 = 2 * err

    if e2 >= -dy:
        err -= dy
        x += sx

    if e2 <= dx:
        err += dx
        y += sy

    x_points.append(x)
    y_points.append(y)

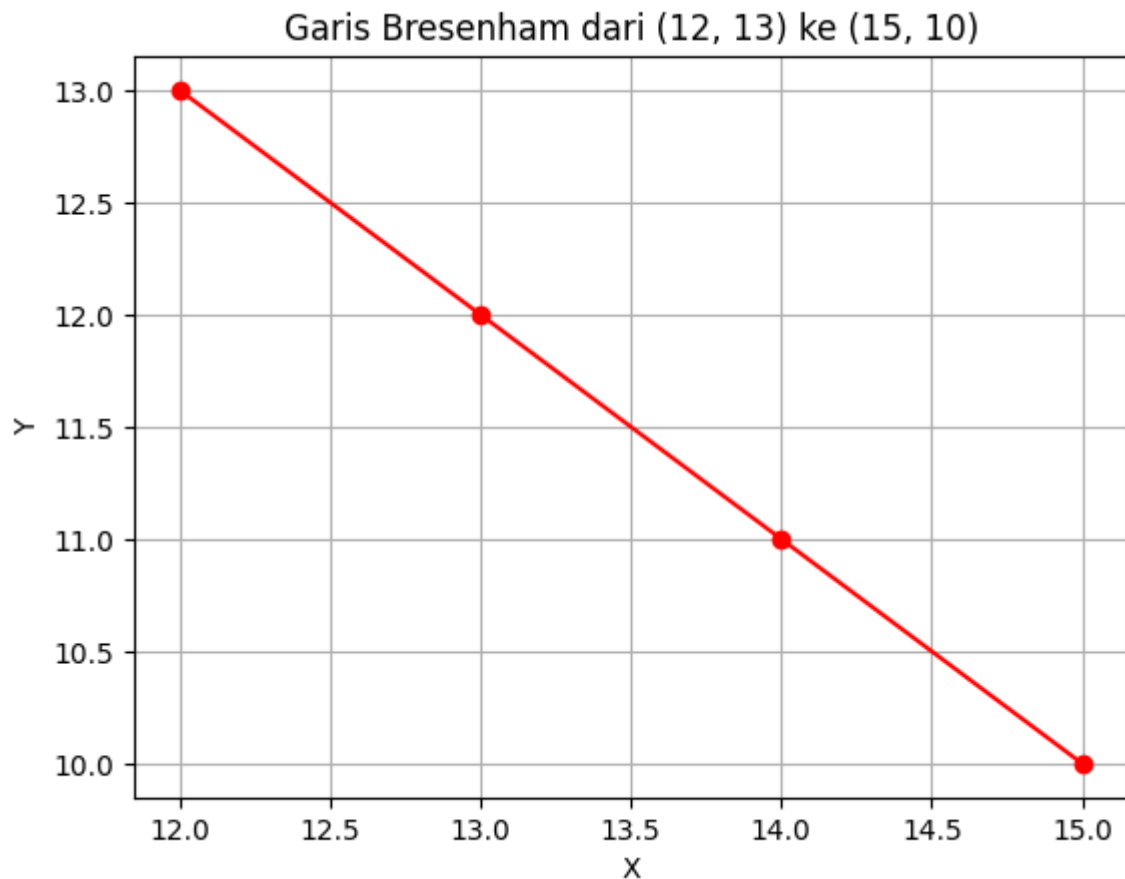
# Menggambar garis menggunakan matplotlib
plt.plot(x_points, y_points, marker="o", color="r")
plt.title(f"Garis Bresenham dari ({x0}, {y0}) ke ({x1}, {y1})")
plt.xlabel("X")
plt.ylabel("Y")
plt.grid(True)
plt.show()

# Meminta input dari pengguna untuk titik awal dan titik akhir
x0 = int(input("Masukkan koordinat x0 (titik awal): "))
y0 = int(input("Masukkan koordinat y0 (titik awal): "))
x1 = int(input("Masukkan koordinat x1 (titik akhir): "))
y1 = int(input("Masukkan koordinat y1 (titik akhir): "))

# Memanggil fungsi untuk menggambar garis dengan algoritma Bresenham
bresenham_algorithm(x0, y0, x1, y1)

```

Hasil Run:



Analisis: Program di atas merupakan implementasi **algoritma Bresenham** untuk menggambar garis antara dua titik pada bidang koordinat menggunakan bahasa Python dan library **matplotlib**. Algoritma Bresenham digunakan untuk menentukan titik-titik koordinat integer pada garis tanpa menggunakan operasi pecahan, sehingga lebih efisien dibandingkan algoritma DDA. Proses diawali dengan menghitung selisih koordinat  $dx$  dan  $dy$ , serta menentukan arah pergerakan garis ( $sx$  dan  $sy$ ) berdasarkan posisi relatif titik awal dan titik akhir. Variabel  $err$  digunakan sebagai parameter kesalahan untuk menentukan kapan koordinat  $y$  perlu diubah selama pergerakan pada sumbu  $x$ . Selama proses iterasi, algoritma akan memperbarui posisi titik  $(x, y)$  dan menyimpannya dalam daftar  $x\_points$  serta  $y\_points$ . Setelah semua titik dihitung, program memvisualisasikan hasilnya dalam bentuk garis berwarna merah dengan marker lingkaran menggunakan fungsi `plt.plot()`. Grafik dilengkapi dengan judul, label sumbu, dan grid agar hasil visualisasi lebih jelas. Program ini juga bersifat interaktif karena meminta pengguna untuk memasukkan titik awal dan titik akhir sebelum menggambar garis.

### 3. Praktikum 3 : Implementasi algoritma Bresenham dengan Input Dinamis

Source Code:

```
import matplotlib.pyplot as plt

def dda_algorithm(x0, y0, x1, y1):
    x, y = x0, y0
```

```

dx = x1 - x0
dy = y1 - y0

# Menentukan steps (langkah)
steps = abs(dx) if abs(dx) > abs(dy) else abs(dy)

# Menghitung inkremen (penambahan per langkah)
x_inc = dx / steps
y_inc = dy / steps

# Inisialisasi list untuk menyimpan koordinat
x_points = [x]
y_points = [y]

# Loop untuk menghitung dan menyimpan titik
for _ in range(int(steps)):
    x += x_inc
    y += y_inc
    x_points.append(round(x))
    y_points.append(round(y))

# Menggambar garis menggunakan matplotlib
plt.plot(x_points, y_points, marker="o", color="b")

def draw_house():
    plt.figure(figsize=(6, 6))

    # Dinding rumah (persegi)
    dda_algorithm(2, 2, 6, 2) # Kiri
    dda_algorithm(2, 6, 6, 6) # Atas
    dda_algorithm(6, 6, 6, 2) # Kanan
    dda_algorithm(6, 2, 2, 2) # Bawah

    # Atap rumah (segitiga)
    dda_algorithm(2, 6, 4, 8) # Kiri ke puncak
    dda_algorithm(4, 8, 6, 6) # Puncak ke kanan

    # Pintu (persegi kecil)
    dda_algorithm(3, 2, 3, 4) # Kiri
    dda_algorithm(3, 4, 4, 4) # Atas
    dda_algorithm(4, 4, 4, 2) # Kanan

    # Jendela (persegi kecil)
    dda_algorithm(5, 5, 5, 6) # Kiri
    dda_algorithm(5, 6, 6, 6) # Atas
    dda_algorithm(6, 6, 6, 5) # Kanan
    dda_algorithm(6, 5, 5, 5) # Bawah

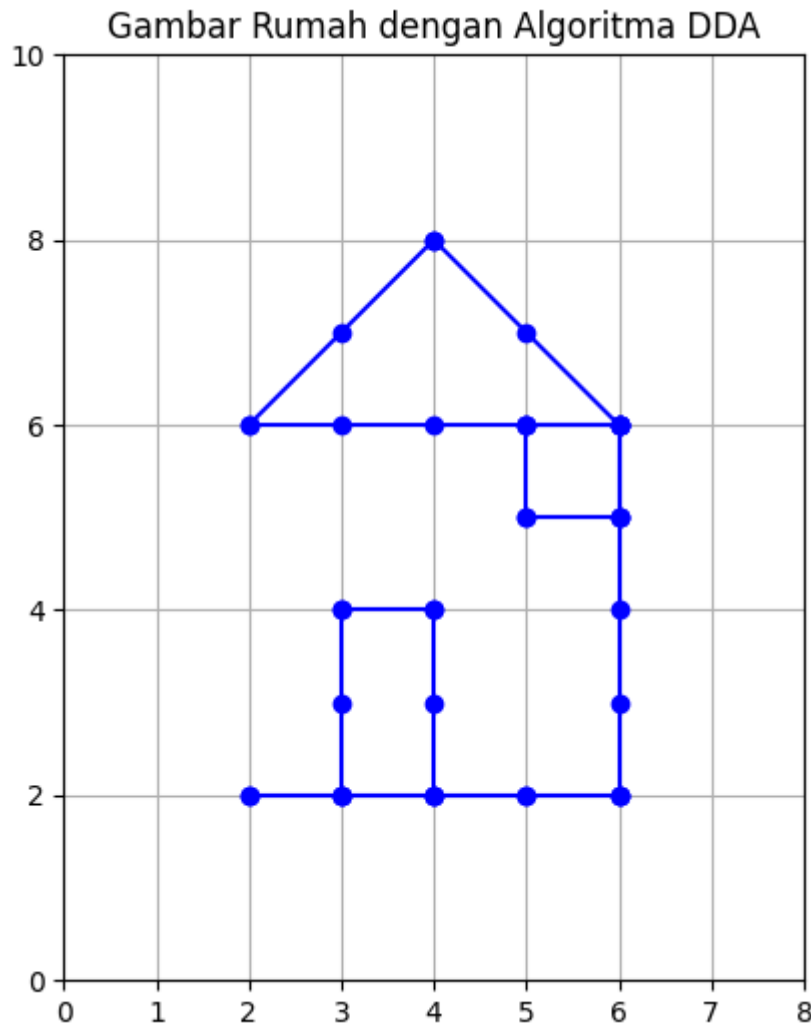
    # Menampilkan hasil gambar rumah
    plt.title("Gambar Rumah dengan Algoritma DDA")
    plt.xlim(0, 8)
    plt.ylim(0, 10)
    plt.gca().set_aspect('equal', adjustable='box')
    plt.grid(True)

```

```
plt.show()

# Memanggil fungsi untuk menggambar rumah
draw_house()
```

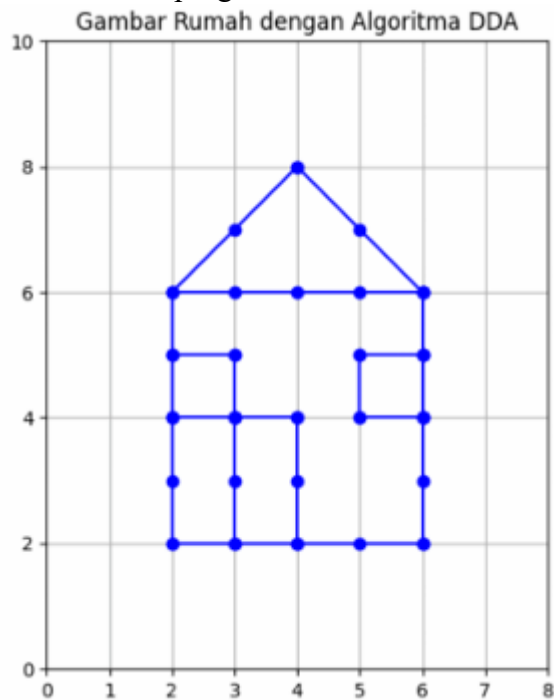
Hasil Run :



Analisis: Program di atas merupakan implementasi **algoritma DDA (Digital Differential Analyzer)** untuk menggambar bentuk rumah sederhana menggunakan bahasa Python dan library **matplotlib**. Algoritma DDA digunakan untuk menggambar garis dengan cara menghitung perubahan koordinat x dan y secara bertahap berdasarkan jumlah langkah (steps) dari perbedaan terbesar antara dx dan dy. Setiap titik garis dihitung dengan menambahkan nilai inkremen ( $x_{inc}$  dan  $y_{inc}$ ) hingga mencapai titik akhir, lalu hasilnya divisualisasikan sebagai titik-titik yang saling terhubung. Pada program ini, fungsi `dda_algorithm()` bertugas menggambar satu garis dari dua titik, sedangkan fungsi `draw_house()` menyusun beberapa garis tersebut menjadi bentuk rumah — terdiri dari **dinding persegi**, **atap segitiga**, **pintu**, dan **jendela**. Hasil akhirnya menampilkan gambar rumah berwarna biru dengan titik-titik tepi yang jelas, lengkap dengan grid dan skala sumbu yang seimbang. Program ini menggambarkan bagaimana algoritma DDA dapat digunakan untuk membentuk objek grafis kompleks melalui kombinasi garis-garis lurus sederhana.

## LATIHAN/TUGAS

1. Buatlah kode program sederhana untuk menggambar rumah seperti gambar berikut:



Source Code:

```
# Tugas
import matplotlib.pyplot as plt
import numpy as np

def dda(x1, y1, x2, y2):
    """
    Implementasi algoritma Digital Differential Analyzer (DDA)
    untuk menghasilkan titik-titik di antara dua koordinat.
    """
    points = []
    dx = x2 - x1
    dy = y2 - y1

    steps = max(abs(dx), abs(dy))

    if steps == 0:
        points.append((round(x1), round(y1)))
        return points

    x_increment = dx / steps
    y_increment = dy / steps

    x = float(x1)
    y = float(y1)

    for _ in range(steps + 1):
        points.append((round(x), round(y)))
        x += x_increment
        y += y_increment
```

```

    return points

# Definisi garis-garis yang membentuk rumah
# Format: ((x1, y1), (x2, y2))
house_lines = [
    # Rangka utama
    ((2, 2), (6, 2)), # Dasar
    ((2, 6), (6, 6)), # Langit-langit
    ((2, 2), (2, 6)), # Dinding kiri
    ((6, 2), (6, 6)), # Dinding kanan
    # Atap
    ((2, 6), (4, 8)), # Atap kiri
    ((4, 8), (6, 6)), # Atap kanan
    # Pintu
    ((3, 2), (3, 4)), # Pintu kiri
    ((4, 2), (4, 4)), # Pintu kanan
    ((3, 4), (4, 4)), # Pintu atas
    # Jendela Kiri
    ((2, 4), (3, 4)),
    ((2, 5), (3, 5)),
    ((2, 4), (2, 5)),
    ((3, 4), (3, 5)),
    # Jendela Kanan
    ((5, 4), (6, 4)),
    ((5, 5), (6, 5)),
    ((5, 4), (5, 5)),
    ((6, 4), (6, 5)),
]

# --- Proses Menggambar ---

# Inisialisasi plot
fig, ax = plt.subplots(figsize=(7, 7))

# Kumpulkan semua titik unik dari algoritma DDA
all_points = set()

# Gambar garis-garis rumah
for line in house_lines:
    (x1, y1), (x2, y2) = line
    # Gambar garis lurus biru yang menghubungkan titik awal dan akhir
    ax.plot([x1, x2], [y1, y2], color='blue', linewidth=1.5)

    # Dapatkan titik-titik dari DDA dan tambahkan ke set
    dda_points = dda(x1, y1, x2, y2)
    for p in dda_points:
        all_points.add(p)

# Gambar titik-titik yang dihasilkan oleh DDA
for x, y in all_points:
    ax.plot(x, y, 'bo') # 'bo' berarti blue circle marker

# --- Pengaturan Tampilan Plot ---

```

```
# Atur judul
ax.set_title("Gambar Rumah dengan Algoritma DDA", fontsize=14)

# Atur batas sumbu
ax.set_xlim(0, 8)
ax.set_ylim(0, 10)

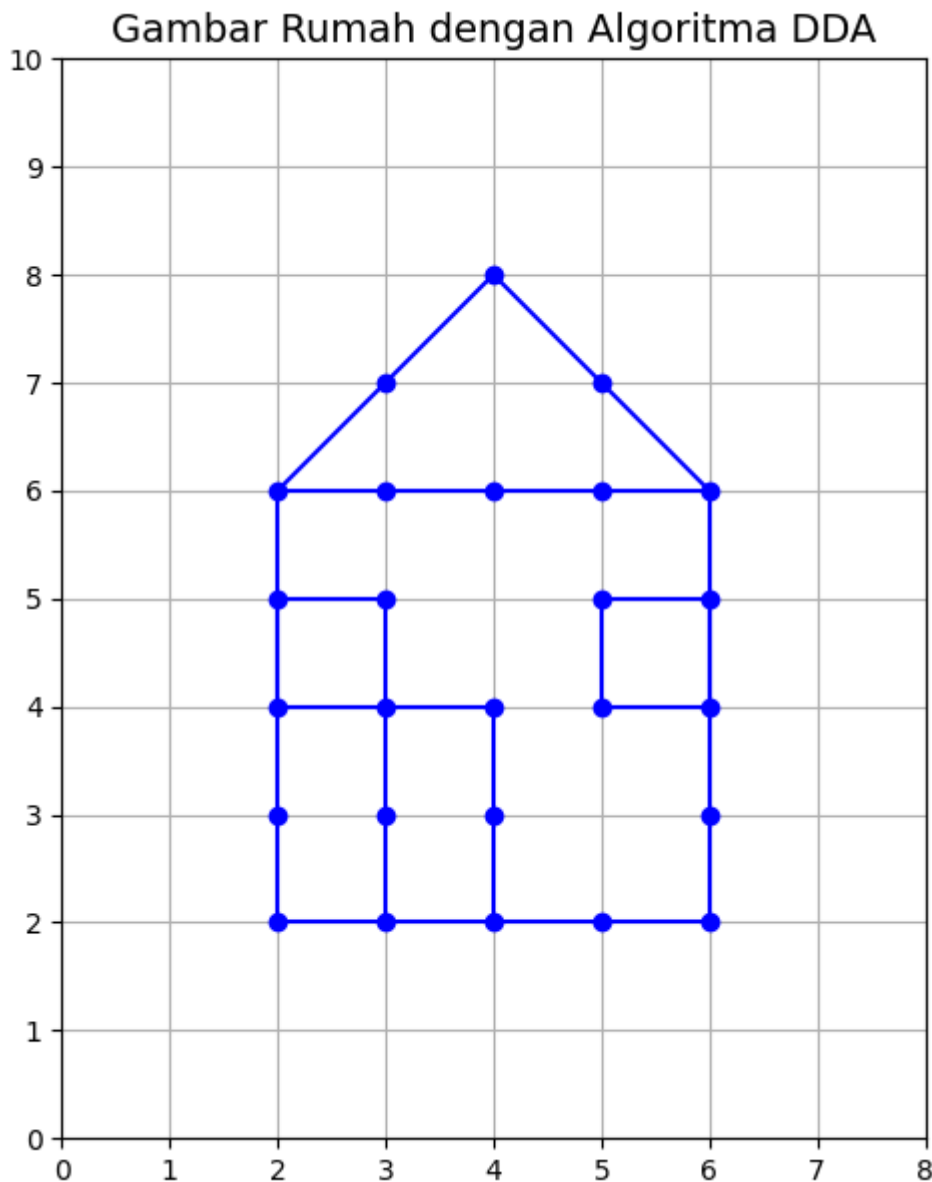
# Atur tick/tanda pada sumbu agar menjadi integer
ax.set_xticks(np.arange(0, 9, 1))
ax.set_yticks(np.arange(0, 11, 1))

# Tampilkan grid
ax.grid(True)

# Pastikan rasio aspek sumbu sama agar gambar tidak terdistorsi
ax.set_aspect('equal', adjustable='box')

# Tampilkan plot
plt.show()
```

Hasil Run :



Analisis: Program Python di atas merupakan implementasi **algoritma Digital Differential Analyzer (DDA)** untuk menggambar bentuk rumah secara grafis menggunakan pustaka **Matplotlib**. Algoritma DDA bekerja dengan menghitung selisih koordinat titik awal ( $x_1, y_1$ ) dan titik akhir ( $x_2, y_2$ ), lalu menentukan jumlah langkah berdasarkan jarak terpanjang antara sumbu X dan Y. Nilai inkremen  $x\_increment$  dan  $y\_increment$  digunakan untuk menambahkan posisi titik secara bertahap, menghasilkan koordinat piksel yang membentuk garis lurus. Dalam program ini, rumah digambar dari kumpulan garis yang membentuk bagian-bagian seperti **dinding**, **atap segitiga**, **pintu**, dan **dua jendela kecil** di sisi kiri dan kanan. Setiap garis diproses dengan fungsi `dda()`, yang menghitung semua titik koordinat di sepanjang garis, kemudian divisualisasikan dengan titik-titik berwarna biru. Hasil akhirnya adalah gambar rumah yang proporsional dengan bentuk jelas, dilengkapi grid dan skala sumbu yang presisi. Program ini menunjukkan bagaimana algoritma DDA dapat digunakan secara efektif untuk

menggambar bentuk geometris kompleks dari kombinasi garis lurus sederhana dalam pemrograman grafis.