

LAPORAN PRAKTIKUM

GRAFIKA KOMPUTER

(DOSEN PENGAMPU : Rio Priantama, S.T., M.T.I)

Modul 4



DISUSUN OLEH :

NAMA: MOHAMAD ABAN SY'BANA

NIM : 20230810012

KELAS : TINFC-2023-04

TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS KUNINGAN

2025

PRAKTIKUM

1. Menggambar Lingkaran dengan Algoritma Midpoint.

Code Program

```
import matplotlib.pyplot as plt

def midpoint_circle(h, k, r):
    # Mulai dari titik teratas pada lingkaran relatif terhadap pusat (0, r)
    x = 0
    y = r
    points = []

    # Parameter keputusan awal untuk algoritma midpoint
    p = 1 - r

    # Tambahkan titik awal (sudah digeser ke pusat (h, k))
    points.append((x + h, y + k))

    # Ulangi sampai x mencapai atau melewati y (hanya men-generate satu oktaf lalu cermin)
    while x < y:
        x += 1 # naikkan x satu langkah
        # Jika p < 0, titik selanjutnya tetap pada y, hanya update p berdasarkan x
        if p < 0:
            p += 2 * x - 2 * + 1
        else:
            # Jika p >= 0, turunkan y satu langkah dan update p berdasarkan perbedaan x dan y
            y -= 1
            p += 2 * x - 2 * y + 1

    # Karena simetri lingkaran, tambahkan 8 titik yang merupakan cermin dari (x, y)
    points.extend([
        (x + h, y + k),
        (h - x, k + y),
        (h + x, k - y),
        (h - x, k - y),
        (h + y, k + x),
        (h - y, k + x),
        (h + y, k - x),
        (h - y, k - x)
    ])
    return points

# Koordinat pusat dan jari-jari lingkaran yang ingin digambar
center_x = 0
center_y = 0
radius = 10

# Dapatkan titik-titik lingkaran dari fungsi midpoint_circle
circle_points = midpoint_circle(center_x, center_y, radius)

# Visualisasi menggunakan matplotlib
plt.figure(figsize=(8,8))
for point in circle_points:
    # Gambar setiap titik sebagai titik biru ('bo')
    plt.plot(point[0], point[1], 'bo')

# Judul plot (diletakkan di dalam loop pada kode asli; cukup satu kali juga bisa)
```

```
plt.title('lingkaran dengan menggunakan algoritma midpoint circle')
```

```
# Atur batas sumbu agar lingkaran terlihat penuh
```

```
plt.xlim(-radius-1, radius+1)
```

```
plt.ylim(-radius-1, radius+1)
```

```
# Gambar garis koordinat (sumbu x dan y) dengan style tipis dan putus-putus
```

```
plt.axhline(0, color='black', linewidth=0.5, ls='--')
```

```
plt.axvline(0, color='black', linewidth=0.5, ls='--')
```

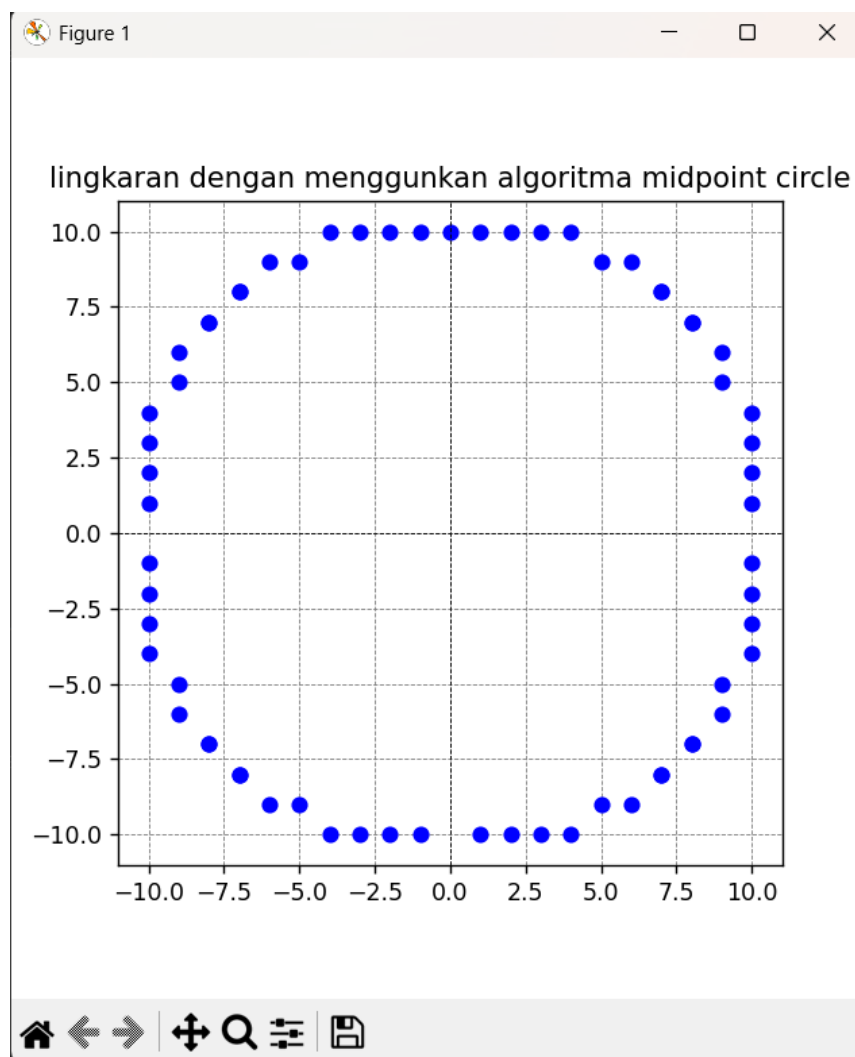
```
# Tampilkan grid dan pastikan aspek sama (agar lingkaran tidak terdistorsi)
```

```
plt.grid(color='gray', linestyle='--', linewidth=0.5)
```

```
plt.gca().set_aspect('equal', adjustable='box')
```

```
plt.show()
```

Hasil RUN



2. Menggambar Lingkaran dengan Algoritma Simetri.

Code Program

```
import matplotlib.pyplot as plt
```

```
def midpoint_circle(h, k, r):
```

```
    """
```

```
    Menghasilkan daftar titik (x, y) pada lingkaran dengan algoritma midpoint circle.
```

```
    h, k: koordinat pusat lingkaran
```

```
    r: jari-jari (radius)
```

```
    """
```

```

x = 0      # mulai dari titik paling kanan pada sumbu x relatif ke pusat (0, r)
y = r      # mulai dari titik paling atas relatif ke pusat
points = []

# Inisialisasi keputusan p sesuai algoritma midpoint circle
p = 1 - r

# Tambahkan titik awal (h + 0, k + r)
points.append((x + h, y + k))

# Iterasi sampai x < y (menggambar sampai sudut 45 derajat, sisanya simetri)
while x < y:
    x += 1
    # Jika p < 0 berarti titik di dalam lingkaran -> gunakan piksel sebelah kanan
    # Pembaruan p untuk kasus ini: p = p + 2*x + 1
    if p < 0:
        p += 2 * x + 1
    else:
        # Jika p >= 0 berarti titik berada di luar/tepat di batas -> geser ke bawah (y-1)
        y -= 1
        # Pembaruan p untuk kasus ini: p = p + 2*(x - y) + 1
        p += 2 * x - 2 * y + 1

# Karena lingkaran simetris terhadap 8 oktaf, tambahkan semua titik simetris
points.extend([
    (h + x, k + y), # kuadran kanan-atas relatif
    (h - x, k + y), # kiri-atas
    (h + x, k - y), # kanan-bawah
    (h - x, k - y), # kiri-bawah
    (h + y, k + x), # rotasi 90 derajat dari (x,y)
    (h - y, k + x),
    (h + y, k - x),
    (h - y, k - x)
])
return points

# Input dari pengguna
center_x = int(input("Masukkan koordinat x pusat lingkaran: "))
center_y = int(input("Masukkan koordinat y pusat lingkaran: "))
radius = int(input("Masukkan jari-jari lingkaran: "))

# Dapatkan daftar titik lingkaran menggunakan fungsi di atas
circle_points = midpoint_circle(center_x, center_y, radius)

# Plot hasilnya menggunakan matplotlib
plt.figure(figsize=(8,8))
for point in circle_points:
    plt.plot(point[0], point[1], 'bo') # 'bo' = blue circle marker

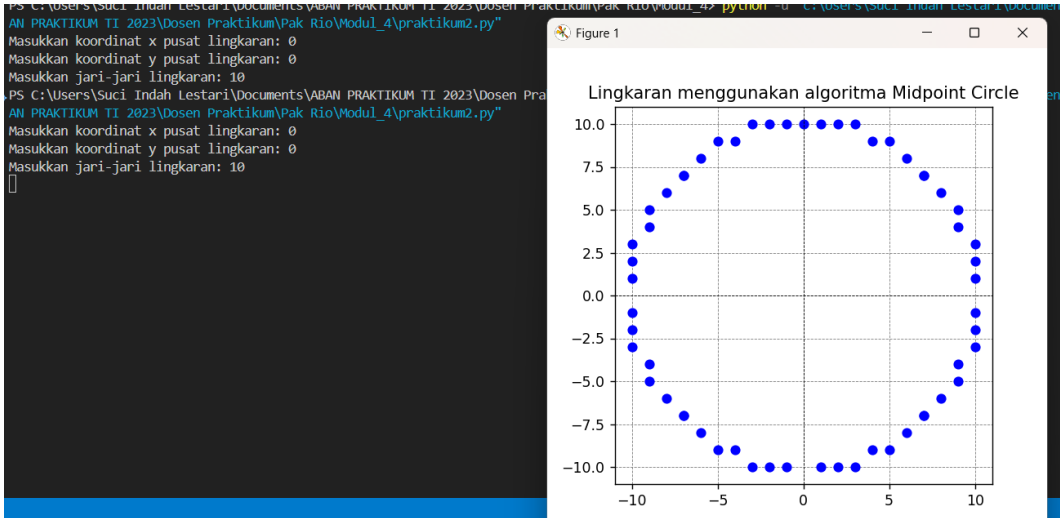
plt.title('Lingkaran menggunakan algoritma Midpoint Circle')
# Atur batas sumbu agar lingkaran muat pada tampilan
plt.xlim(center_x - radius - 1, center_x + radius + 1)
plt.ylim(center_y - radius - 1, center_y + radius + 1)

# Tambahkan sumbu bantu dan grid
plt.axhline(0, color='black', linewidth=0.5, ls='--')
plt.axvline(0, color='black', linewidth=0.5, ls='--')
plt.grid(color='gray', linestyle='--', linewidth=0.5)

```

```
plt.gca().set_aspect('equal', adjustable='box') # aspek rasio 1:1 agar lingkaran tidak terdistorsi
plt.show()
```

Hasil RUN



3. Menggambar Lingkaran dengan Algoritma Bresenhem.

Code Program

```
import matplotlib.pyplot as plt

def plot_circle(x_center, y_center, x, y):
    """
    Gambarkan titik-titik simetris untuk satu pasangan koordinat (x, y)
    relatif terhadap pusat lingkaran (x_center, y_center).

    Param:
    - x_center, y_center: koordinat pusat lingkaran
    - x, y: koordinat titik pada kuadran pertama yang akan dipantulkan
    """
    # Gambar 8 titik simetris berdasarkan pasangan (x, y)
    plt.plot(x_center + x, y_center + y, 'ro')
    plt.plot(x_center - x, y_center + y, 'ro')
    plt.plot(x_center + x, y_center - y, 'ro')
    plt.plot(x_center - x, y_center - y, 'ro')
    plt.plot(x_center + y, y_center + x, 'ro')
    plt.plot(x_center - y, y_center + x, 'ro')
    plt.plot(x_center + y, y_center - x, 'ro')
    plt.plot(x_center - y, y_center - x, 'ro')
```

```
def bresenham_circle(x_center, y_center, radius):
    """
    Implementasi algoritma Bresenham untuk menggambar lingkaran.
```

Algoritma ini menggunakan parameter keputusan 'd' untuk menentukan apakah koordinat y harus dikurangi pada langkah berikutnya.
Fungsi memanggil plot_circle pada setiap langkah untuk menggambar semua titik simetris lingkaran.

Param:

- x_center, y_center: koordinat pusat lingkaran
- radius: jari-jari lingkaran (integer)

"""

Titik awal pada sumbu y (x = 0, y = radius)

```

x = 0
y = radius
# Nilai awal fungsi keputusan (konstanta untuk lingkaran)
d = 3 - 2 * radius

# Gambar titik awal (beserta simetrisnya)
plot_circle(x_center, y_center, x, y)

# Iterasi sampai y menjadi negatif (selesai menggambar kuadran)
while y >= 0:
    x += 1
    # Jika d > 0, geser y ke dalam (decrement) dan perbarui d sesuai kasus
    if d > 0:
        y -= 1
        d = d + 4 * (x - y) + 10
    else:
        # Jika d <= 0, hanya perbarui d tanpa mengubah y
        d = d + 4 * x + 6
    # Gambar titik hasil iterasi saat ini (beserta simetrisnya)
    plot_circle(x_center, y_center, x, y)

# Input pengguna untuk pusat lingkaran dan jari-jari
x_center = int(input("Masukkan koordinat x pusat lingkaran: "))
y_center = int(input("Masukkan koordinat y pusat lingkaran: "))
radius = int(input("Masukkan jari-jari lingkaran: "))

# Panggil fungsi untuk menggambar lingkaran
bresenham_circle(x_center, y_center, radius)

# Atur aspek, tampilkan grid, dan tampilkan plot
plt.gca().set_aspect('equal', adjustable='box')
plt.grid(True)
plt.show()

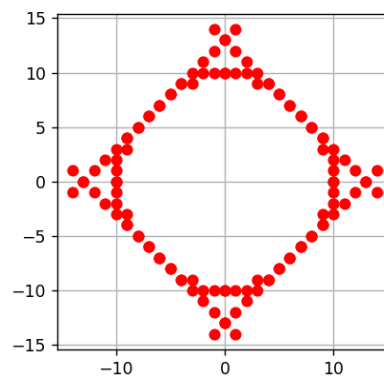
```

Hasil RUN

```

PS C:\Users\Suci Indah Lestari\Documents\ABAN PRAKTIKUM T
023\Dosen Praktikum\Pak Rio\Modul_4\praktikum3.py"
Masukkan koordinat x pusat lingkaran: 0
Masukkan koordinat y pusat lingkaran: 0
Masukkan jari-jari lingkaran: 10

```



TUGAS

1. Buatlah kode sederhana untuk menggambar lingkaran seperti gambar berikut :

Code Program

```
import matplotlib.pyplot as plt

def midpoint_circle(h, k, r):
    x = 0
    y = r
    points = set()

    p = 1 - r

    # tambahkan semua 8 simetri untuk titik awal
    points.update([
        (x + h, y + k),
        (h - x, k + y),
        (h + x, k - y),
        (h - x, k - y),
        (h + y, k + x),
        (h - y, k + x),
        (h + y, k - x),
        (h - y, k - x)
    ])

    while x < y:
        x += 1
        if p < 0:
            p += 2 * x + 1
        else:
            y -= 1
            p += 2 * (x - y) + 1

        points.update([
            (x + h, y + k),
            (h - x, k + y),
            (h + x, k - y),
            (h - x, k - y),
            (h + y, k + x),
            (h - y, k + x),
            (h + y, k - x),
            (h - y, k - x)
        ])

    return list(points)

center_x = 0
center_y = 0
radius = 10

circle_points = midpoint_circle(center_x, center_y, radius)

# ubah titik menjadi dua list untuk plotting
xs = [p[0] for p in circle_points]
ys = [p[1] for p in circle_points]

plt.figure(figsize=(8,8))
plt.scatter(xs, ys, s=30, color='blue') # gunakan scatter untuk kontrol ukuran titik

# set padding per sisi (kanan, kiri, atas, bawah)
```

```

pad_right = 0.2
pad_left = 0.2
pad_top = 0.2
pad_bottom = 0.2

x_min = center_x - radius - pad_left
x_max = center_x + radius + pad_right
y_min = center_y - radius - pad_bottom
y_max = center_y + radius + pad_top

ax = plt.gca()
ax.set_xlim(x_min, x_max)
ax.set_ylim(y_min, y_max)

plt.title('lingkaran dengan menggunakan algoritma midpoint circle')
plt.axhline(0, color='black', linewidth=0.5, ls='--')
plt.axvline(0, color='black', linewidth=0.5, ls='--')
plt.grid(color='gray', linestyle='--', linewidth=0.5)
ax.set_aspect('equal', adjustable='box')
plt.show()

```

Hasil RUN

