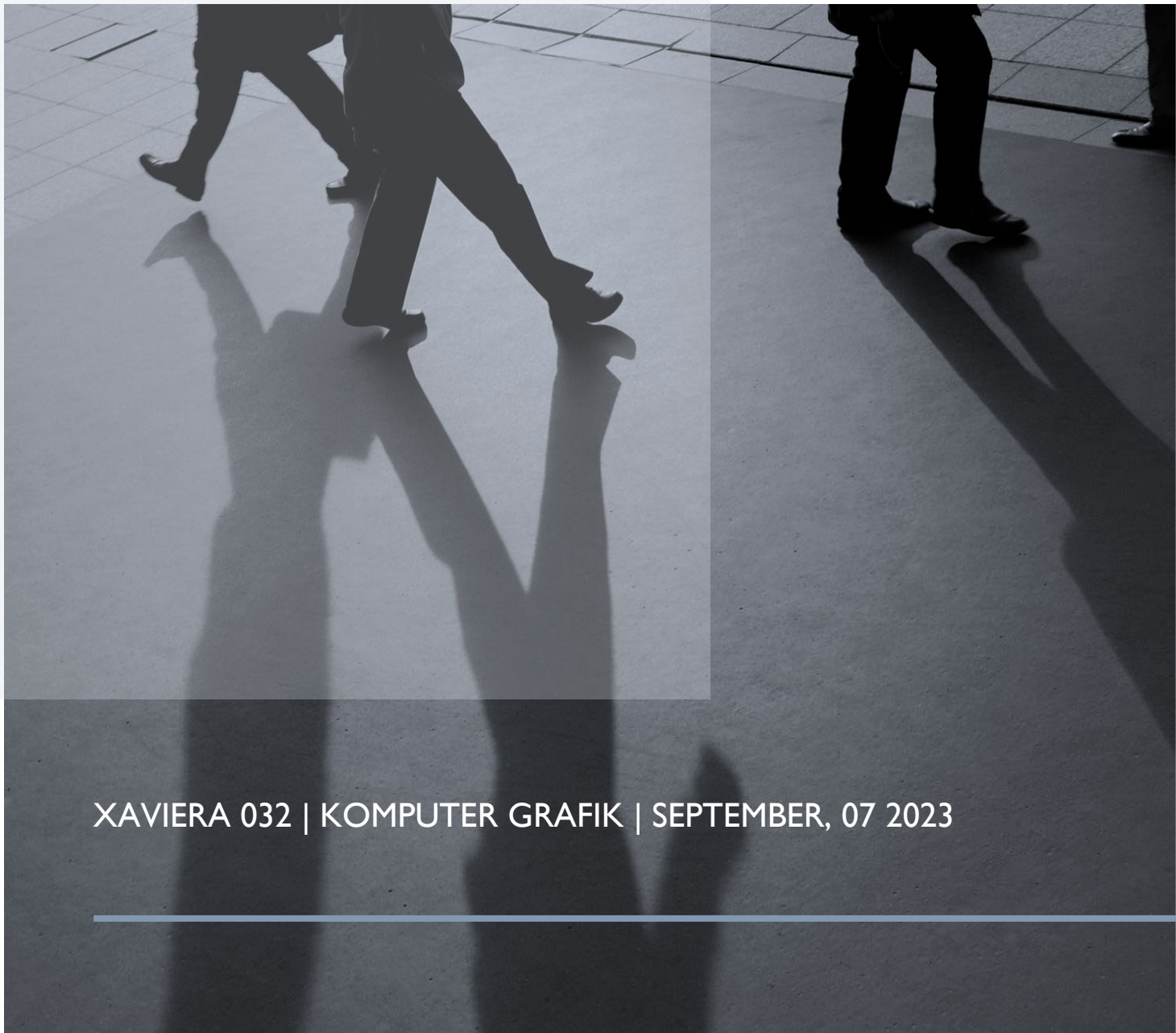
MODUL III

KOMPUTER GRAFIK 2D
LINGKARAN DAN ELLIPS

D3 TEKNIK INFORMATIKA
JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA
POLITEKNIK NEGERI BANDUNG



XAVIERA 032 | KOMPUTER GRAFIK | SEPTEMBER, 07 2023

CONTENTS

LINGKARAN

Lingkaran merupakan bentuk dasar yang biasa digunakan untuk membuat gambar atau objek yang kompleks, seperti dekorasi, batik, dan lain-lain. Pada paket library komputer grafik, sebagian atau lingkaran penuh dapat dibuat menggunakan sebuah prosedur. Secara general, prosedur tersebut dapat menghasilkan garis dan ellips.

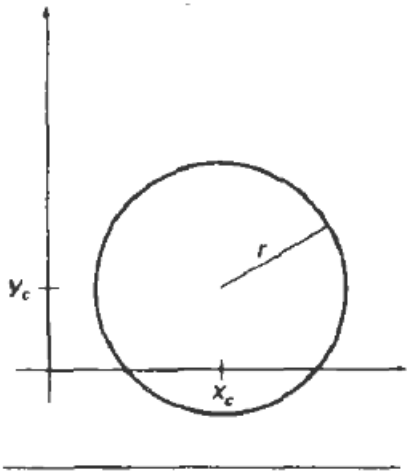


Figure 3-12
Circle with center coordinates (x_c, y_c) and radius r.

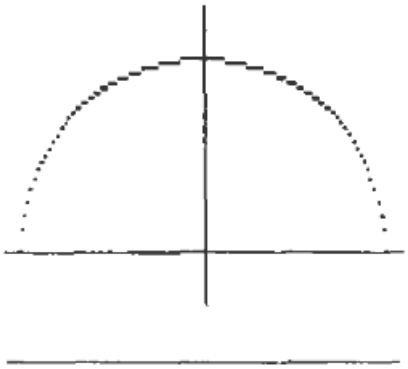


Figure 3-13
Positive half of a circle plotted with Eq. 3-25 and with $(x_c, y_c) = (0, 0)$.

Persamaan Lingkaran

$$(x - x_c)^2 + (y - y_c)^2 = r^2$$

$$y = y_c \pm \sqrt{(r^2 - (x_c - x)^2)}$$

Persamaan Lingkaran Polar Coordinates

$$x = x_c + r\ cos\ \theta$$

$$y = y_c + r\ sin\ \theta$$

Step size :

$$\frac{1}{r}$$

Komputasi lingkaran dapat direduksi karena lingkaran adalah bentuk yang simetris. Setiap kuadran meliliki bentuk bagian lingkaran yang sama.

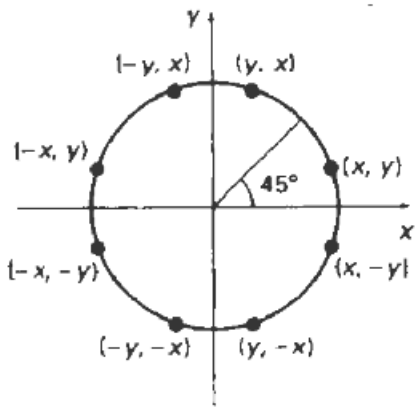


Figure 3-14
Symmetry of a circle. Calculation of a circle point (x, y) in one octant yields the circle points shown for the other seven octants.

Algoritma paling efisien berdasarkan kalkulasi incremental dari decision parameter, seperti bersenham line algorithm, yang hanya membutuhkan operasi integer sederhana. Algoritma line bersenham diadaptasi untuk pembentukan lingkaran dengan melakukan setup decision parameter untuk mencari pixel terdekat untuk mendapatkan lingkar untuk setiap sampling step.

Metode untuk mendapatkan jarak secara langsung pada lingkaran, adalah dengan melakukan pengecekan posisi tengah dari dua pixel untuk menentukan apakah titik "midpoint" ini ada pada dalam atau luar lingkaran. Metode midpoint dapat diaplikasikan untuk bentuk-bentuk conics.

MIDPOINT CIRCLE ALGORITHM

Seperti pada line bersenham, tujuan dari algoritma untuk mendapatkan sampling titik dan menentukan pixel terdekat pada setiap step. Pada lingkaran, untuk setiap r dan titik posisi center (x_c, y_c). Algoritma dimulai dengan mengkalkulasi posisi pixel dalam jalur lingkaran yang memiliki titik tengah origin $(0,0)$. Lalu untuk setiap posisi (x, y) yang dikalkulasi dipindahkan pada posisi yang benar dengan menambahkan $x_c$pada x dan y_c pada y.

Pada bagian lingkaran kuadran I mulai dari $x = 0, x = y$ nilai slope bervariasi dari 0 sampai -1 . Maka pergerakan unit step sesuai arah x positif dan menggunakan decision parameter untuk menentukan dua posisi yang mungkin lebih dekat dengan jalur lingkaran untuk setiap langkah.

$$f_{circle}(x, y) = x^2 + y^2 - r^2$$

Jika point ada didalam interior lingkaran , fungsi lingkaran negative, dan sebaliknya.

$$f\_circle(x, y) = \begin{cases} < 0, & if\ (x, y)\ is\ inside\ the\ circle\ boundary \\ = 0, & if\ (x, y)\ is\ on\ the\ circle\ boundary \\ > 0, & if\ (x, y)\ is\ outside\ the\ circle\ boundary \end{cases}$$

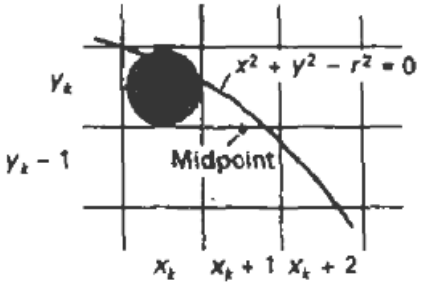


Figure 3-15
Midpoint between candidate pixels at sampling position x_k+1 along a circular path.

$$p_k = f_{circle}\left(x_k + 1, y_k - \frac{1}{2}\right)$$

$$= (x+1)^2 + \left(y_k - \frac{1}{2}\right)^2 - r^2$$

Jika $p_k < 0$ midpoint terletak pada bagian dalam lingkaran dan y_k lebih dekat pada batas lingkaran. Sebaliknya $y_k - 1$ lebih dekat pada batas lingkaran.

Decision parameter selanjutnya didapatkan menggunakan kalkulasi incremental integer, yaitu $x_{k+1} + 1 = x_k + 2$

$$p_{k+1} = f_{circle}\left(x_{k+1} + 1, y_{k+1} - \frac{1}{2}\right)$$

$$= [(x_k + 1) + 1]^2 + \left(y_{k+1} - \frac{1}{2}\right)^2 - r^2$$

$$p_{k+1} = p_k + 2(x_k + 1) + \left(y_{k+1}^2 - y_k^2\right) - (y_{k+1} - y_k) + 1$$

Dimana, y_{k+1} antara y_k atau y_{k-1} , bergantung pada tanda dari p_k

Evaluasi $2x_{k+1}$ dan $2y_{k+1}$ didapatkan secara inkemental menggunakan.

$$2x_{k+1} = 2x_k + 2$$

$$2y_{k+1} = 2y_k - 2$$

Pada titik awal $(0, r)$, decision parameter pertama didapatkan dengan melakukan evaluasi pada fungsi lingkaran dengan nilai $(x_0, y_0) = (0, r)$:

$$p_o = f_{circle}\left(1, r - \frac{1}{2}\right)$$

$$= 1 + \left(\mathrm{r} - \frac{1}{2}\right)^2 - \mathrm{r}^2$$

$$= \frac{5}{4} - r$$

Midpoint Circle Algorithm

1. Input radius r and circle center (x_c, y_c), and obtain the first point on the circumference of a circle centered on the origin as

$$(x_0, y_0) = (0, r)$$

2. Calculate the initial value of the decision parameter as

$$p_0 = \frac{5}{4} - r$$

3. At each x_k position, starting at $k = 0$, perform the following test: If $p_k < 0$, the next point along the circle centered on $(0, 0)$ is (x_{k+1}, y_k) and

$$p_{k+1} = p_k + 2x_{k+1} + 1$$

   Otherwise, the next point along the circle is $(x_k + 1, y_k - 1)$ and

$$p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}$$

   where $2x_{k+1} = 2x_k + 2$ and $2y_{k+1} = 2y_k - 2$.

4. Determine symmetry points in the other seven octants.
5. Move each calculated pixel position (x, y) onto the circular path centered on (x_c, y_c) and plot the coordinate values:

$$x = x + x_c, \qquad y = y + y_c$$

6. Repeat steps 3 through 5 until $x \geq y$.

```
#include "device.h"

void circleMidpoint (int xCenter, int yCenter, int radius)
{
  int x = 0;
  int y = radius;
  int p = 1 - radius;
  void circlePlotPoints (int, int, int, int);

  /* Plot first set of points */
  circlePlotPoints (xCenter, yCenter, x, y);

  while (x < y) {
    x++;
    if (p < 0)
      p += 2 * x + 1;
    else {
      y--;
      p += 2 * (x - y) + 1;
    }
    circlePlotPoints (xCenter, yCenter, x, y);
  }
}

void circlePlotPoints (int xCenter, int yCenter, int x, int y)
{
  setPixel (xCenter + x, yCenter + y);
  setPixel (xCenter - x, yCenter + y);
  setPixel (xCenter + x, yCenter - y);
  setPixel (xCenter - x, yCenter - y);
  setPixel (xCenter + y, yCenter + x);
  setPixel (xCenter - y, yCenter + x);
  setPixel (xCenter + y, yCenter - x);
  setPixel (xCenter - y, yCenter - x);
}
```

Implementasi Circle Algorithm pada py5

Contoh Pemanggilan Lingkaran menggunakan Points (kumpulan titik)
py5.stroke(0,randint(0,255),0,255) py5.points( )

ELLIPS

Untuk menggambarkan sebuah ellips, kita dapat mengadopsi pola penggambaran lingkaran. Ellips bisa juga disebut lingkaran yang pipih, dengan modifikasi lingkaran yang memiliki dimensi vertikal dan horizontal yang berbeda atau disebut major dan minor axes.

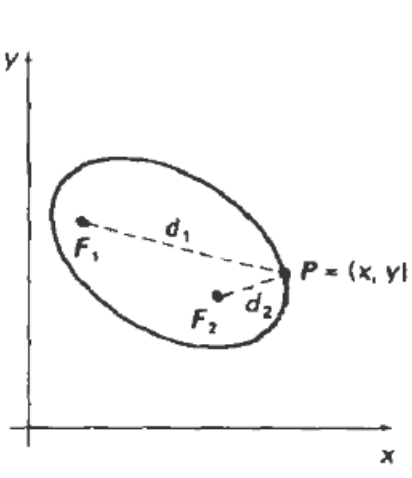


Figure 3-17
Ellipse generated about foci F_1 and F_2.

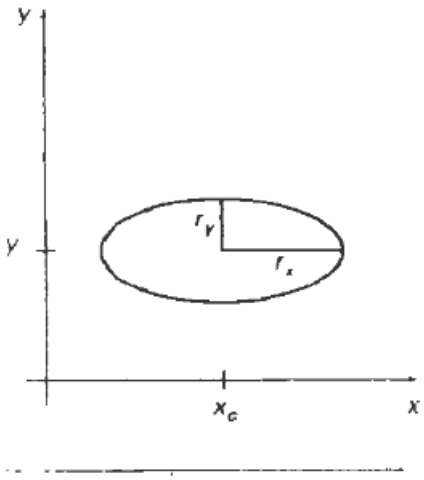


Figure 3-18
Ellipse centered at (x_c, y_c) with semimajor axis r_x and semiminor axis r_y.

Ellips dapat didefinisikan sebagai Kumpulan dari titik yang sedemikian hingga jumlah dari jarak antara dua titik tetap (foci atau fixed position) adalah sama untuk seluruh titik. Jika jarak dari dua foci dari sebuah titik $P = (x, y)$ pada sebuah ellips diberi label d_1 dan d_2, maka persamaan ellips dapat dinyatakan sebagai:

$$d_1 + d_2 = constant$$

Eksprsi dari jarak d1 dan d2 dalam foci $F_1 = (x_1, y_1)$ dan $F_2 = (x_2, y_2)$ maka didapatkan :

$$\sqrt{((x - x_1)^2 + (y - y_1)^2)} + \sqrt{((x - x_2)^2 + (y - y_2)^2)} = constant$$

Persamaan ellips lain ketika posisi mayor axes dan minor axes pada posisi standar dapat dinyatakan sebagai:

$$\left(\frac{x - x_c}{r_x}\right)^2 + \left(\frac{y - y_c}{r_y}\right)^2 = 1$$

Dengan menggunakan koordinat polar ellips pada posisi standar dapat dinyatakan dengan:

$$x = x_c + r_x cos\ \theta$$

$$y = y_c + r_y sin\ \theta$$

MIDPOINT ELLIPS ALGORITHM

Seperti pada penggambaran lingkaran, The digital differential analyzer (DDA) adalah algoritma line scan-conversion berdasarkan dari perhitungan antara dy atau dx pada persamaan garis. Proses sampling pada satu unit koordinat (x atau y) dan menentukan nilai integer terdekat yang sesuai dengan jalur garis dari koordinat lain.

Metode midpoint ellips diaplikasikan pada quadran I untuk dua bagian yaitu Region I (Mayor axes) dan Region II (Minor Axis)

Steps atau iterasi dilakukan pada arah x jika $m < 1$ sebaliknya iterasi dilakukan pada arah y jika $m > 1$. Titik awal pada posisi $(0, r_y)$ dan arah mengikuti jarum jam atau (CW) pada kuadran I ellips. Perubahan unit step dari x ke unit step y ketika $m < -1$. Lalu, mirip dengan lingkaran ada property simetris pada penggambaran ellips, secara parallel untuk dua region.

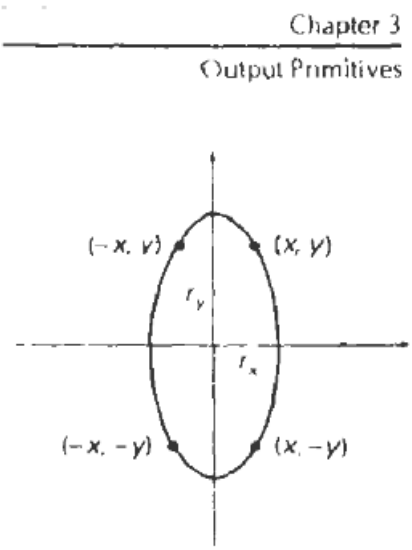


Figure 3-19
Symmetry of an ellipse
Calculation of a point (x, y) in one quadrant yields the ellipse points shown for the other three quadrants.

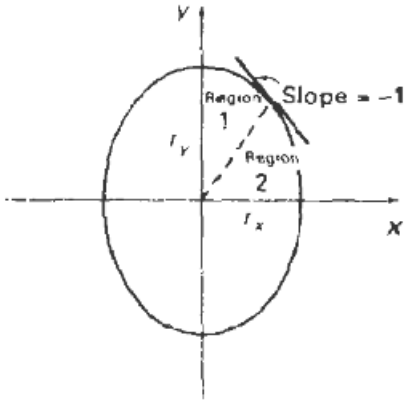


Figure 3-20
Ellipse processing regions. Over region 1, the magnitude of the ellipse slope is less than 1; over region 2, the magnitude of the slope is greater than 1.

Didefinisikan fungsi ellips pada sebuah titik $(x_c, y_c) = (0,0)$ sebagai berikut:

$$f_{ellipse}(x, y) = r_y^2 x^2 + r_x^2 y^2 - r_x^2 r_y^2$$

Yang mengikuti syarat-syarat sebagai berikut:

$$f_{ellipse}(x, y) \begin{cases} < 0, if(x, y)\ is\ inside\ the\ ellipse\ boundary \\ = 0, if(x, y) is\ on\ the\ ellipse\ boundary \\ > 0, if(x, y) is\ outside\ the\ ellipse\ boundary \end{cases}$$

fungsi ellipse berfungsi sebagai decision parameter pada algoritma midpoint. Untuk setiap posisi sampling, pemilihan pixel selanjutnya pada jalur ellips tergantung pada kondisi dari fungsi ellips yang dievaluasi pada midpoint untuk dua kandidat pixel.

Ellipse slope atau m didapatkan dari :

$$\frac{dy}{dx} = -\frac{2r_y^2}{2r_x^2}$$

Pada boundary antara region 1 dan region 2, nilai dari $\frac{dy}{dx} = -1$ dan $2r_y^2 x = 2r_x^2 y$

Maka, pergantian region terjadi saat

$$2r_y^2 x \geq 2r_x^2 y$$

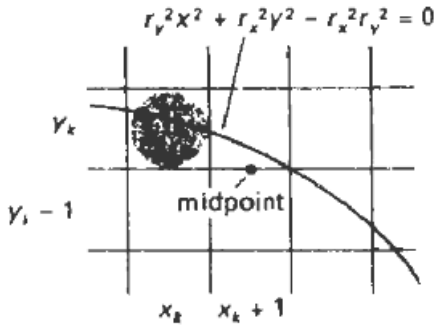


Figure 3-21
Midpoint between candidate pixels at sampling position $x_k + 1$ along an elliptical path.

$$p1_k = f_{ellipse}\left(x_k + 1, y_k - \frac{1}{2}\right)$$

$$= r_y^2(x_k + 1)^2 + r_x^2\left(y_k - \frac{1}{2}\right) - r_x^2 r_y^2$$

$$y_{k+1} = \begin{cases} jika\ p1_k < 0, y_k \\ jika\ p1_k > 0, y_k - 1 \end{cases}$$

$$p1_{k+1} = f_{ellipse}\left(x_{k+1} + 1, y_{k+1} - \frac{1}{2}\right)$$

$$= r_y^2[(x_k + 1) + 1]^2 + r_x^2\left(y_{k+1} - \frac{1}{2}\right)^2 - r_x^2 r_y^2$$

Atau di expand sbb:

$$p1_{k+1} = p1_k + 2r_y^2(x_k + 1) + r_y^2 + r_x^2\left[\left(y_{k+1} - \frac{1}{2}\right)^2 - \left(y_k - \frac{1}{2}\right)^2\right]$$

Subsitusi y_{k+1}antara y_k atau $y_k - 1$ bergantung pada tanda $p1_k$, decision parameter di inkrement sebagai berikut:

$$incerement = \begin{cases} 2r_y^2 x_{k+1} + r_y^2, jika\ p1_k < 0 \\ 2r_y^2 x_{k+1} + r_y^2 - 2r_x^2 y_{k+1}, jika\ p1_k > 0 \end{cases}$$

Pada region 1 nilai pertama pada decision paramater didapatakn dengan mengevaluasi fungsi ellips pada posisi awal $(x_0, y_0) = (0, r_y)$

$$p1_0 = f_{ellipse}\left(1, r_y - \frac{1}{2}\right)$$

$$= r_y^2 + r_x^2\left(r_y - \frac{1}{2}\right)^2 - r_x^2 r_y^2$$

Atau

$$p1_0 = r_y^2 - r_x^2 r_y^2 + \frac{1}{4}r_x^2$$

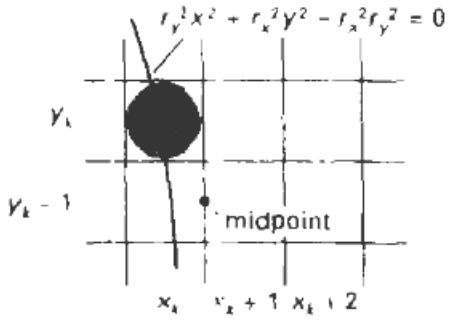


Figure 3-22
Midpoint between candidate pixels at sampling position $y_k - 1$ along an elliptical path.

Untuk region 2, sampling unit step terjadi pada arah y negatif, dan midpoint diambil diantara dua titik kandidat horizontal pixel untuk setiap step. Maka decision parameter dievaluasi sbb:

$$p2_k = f_{ellipse}\left(x_k + \frac{1}{2}, y_k - 1\right)$$

$$= r_y^2\left(x_k + \frac{1}{2}\right)^2 + r_x^2(y_k - 1)^2 - r_x^2 r_y^2$$

$$\mathrm{x_{k+1}} = \begin{cases} jika\ p2_k > 0, x_k \\ jika\ p2_k \leq\ 0, x_{k+1} \end{cases}$$

$$p2_{k+1} = f_{ellipse}\left(x_{k+1} + \frac{1}{2}, y_{k+1} - 1\right)$$

$$= r_y^2\left[x_{k+1} + \frac{1}{2}\right]^2 + r_x^2[(y_k - 1) \quad - 1]^2 - r_x^2 r_y^2$$

Atau

$$p2_{k+1} = p2_k + 2r_x^2(y_k - 1) + r_x^2 + r_y^2\left[\left(x_{k+1} + \frac{1}{2}\right)^2 - \left(x_k + \frac{1}{2}\right)^2\right]$$

Para region 2, nilai pertama adalah (x_0, y_o) diambil berdasarkan posisi terakhir dari region 1 dan dicision parameter pertama region ke 2 adalah.

$$p2_0 = f_{ellipse}\left(x_0 + \frac{1}{2}, y_0 - 1\right)$$

$$= r_y^2\left(x_0 + \frac{1}{2}\right)^2 + r_x^2(y_0 - 1)^2 - r_x^2 r_y^2$$

Untuk menyederhanakan perhitungan, titik pertama dapat diambil pada $(r_x, 0)$. Unit step akan berjalan dengan arah y positif sampai dengan posisi terakhir dari Region 1.

Midpoint Ellipse Algorithm

1. Input r_x, r_y, and ellipse center (x_c, y_c), and obtain the first point on an ellipse centered on the origin as

$$(x_0, y_0) = (0, r_y)$$

2. Calculate the initial value of the decision parameter in region 1 as

$$p1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2$$

3. At each x_k position in region 1, starting at $k = 0$, perform the following test: If $p1_k < 0$, the next point along the ellipse centered on (0, 0) is (x_{k+1}, y_k) and

$$p1_{k+1} = p1_k + 2r_y^2 x_{k+1} + r_y^2$$

Otherwise, the next point along the circle is $(x_k + 1, y_k - 1)$ and

$$p1_{k+1} = p1_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_y^2$$

with

$$2r_y^2 x_{k+1} = 2r_y^2 x_k + 2r_y^2, \qquad 2r_x^2 y_{k+1} = 2r_x^2 y_k - 2r_x^2$$

and continue until $2r_y^2 x \geq 2r_x^2 y$.

and continue until $2r_y^2 x \geq 2r_x^2 y$.

4. Calculate the initial value of the decision parameter in region 2 using the last point (x_0, y_0) calculated in region 1 as

$$p2_0 = r_y^2 \left(x_0 + \frac{1}{2}\right)^2 + r_x^2 (y_0 - 1)^2 - r_x^2 r_y^2$$

5. At each y_k position in region 2, starting at $k = 0$, perform the following test: If $p2_k > 0$, the next point along the ellipse centered on $(0, 0)$ is $(x_k, y_k - 1)$ and

$$p2_{k+1} = p2_k - 2r_x^2 y_{k+1} + r_x^2$$

Otherwise, the next point along the circle is $(x_k + 1, y_k - 1)$ and

$$p2_{k+1} = p2_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_x^2$$

using the same incremental calculations for x and y as in region 1.

6. Determine symmetry points in the other three quadrants.
7. Move each calculated pixel position (x, y) onto the elliptical path centered on (x_c, y_c) and plot the coordinate values:

$$x = x + x_c \qquad y = y + y_c$$

8. Repeat the steps for region 1 until $2r_y^2 x \geq 2r_x^2 y$.

```
#include "device.h"

#define ROUND(a) ((int)(a+0.5))

void ellipseMidpoint (int xCenter, int yCenter, int Rx, int Ry)
{
  int Rx2 = Rx*Rx;
  int Ry2 = Ry*Ry;
  int twoRx2 = 2*Rx2;
  int twoRy2 = 2*Ry2;
  int p;
  int x = 0;
  int y = Ry;
  int px = 0;
  int py = twoRx2 * y;
  void ellipsePlotPoints (int, int, int, int);

  /* Plot the first set of points */
  ellipsePlotPoints (xCenter, yCenter, x, y);

  /* Region 1 */
  p = ROUND (Ry2 - (Rx2 * Ry) + (0.25 * Rx2));
  while (px < py) {
    x++;
    px += twoRy2;
    if (p < 0)
      p += Ry2 + px;
    else {
      y--;
      py -= twoRx2;
      p += Ry2 + px - py;
    }
    ellipsePlotPoints (xCenter, yCenter, x, y);
  }

  /* Region 2 */
  p = ROUND (Ry2*(x+0.5)*(x+0.5) + Rx2*(y-1)*(y-1) - Rx2*Ry2);
  while (y > 0) {
    y--;
    py -= twoRx2;
    if (p > 0)
      p += Rx2 - py;
    else {
      x++;
      px += twoRy2;
      p += Rx2 - py + px;
```

```
    }
    ellipsePlotPoints (xCenter, yCenter, x, y);
  }
}

void ellipsePlotPoints (int xCenter, int yCenter, int x, int y)
{
  setPixel (xCenter + x, yCenter + y);
  setPixel (xCenter - x, yCenter + y);
  setPixel (xCenter + x, yCenter - y);
  setPixel (xCenter - x, yCenter - y);
}
```

Implementasi Ellips Algorithm pada py5

Contoh Pemanggilan Ellips menggunakan Points (kumpulan titik)
py5.stroke(0,randint(0,255),0,255) py5.points( )

TASK PRAKTIKUM

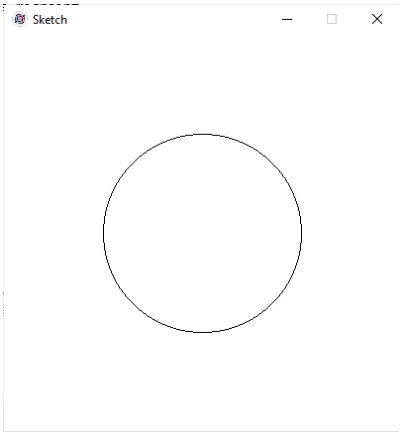
TASK 0-1: IMPLEMENTASI MIDPOINT LINGKARAN DAN MIDPOINT GARIS

1. Lanjutkan kode minggu lalu [KG2023_2X_001_D3_2022]_Modul2, ubah menjadi [KG2023_2X_001_D3_2022]_Modul3
2. Amati Algoritma Midpoint untuk menggambar lingkaran dan Garis
3. Implementasi Algoritma tersebut dan buatlah fungsi lingkaran / circle dan ellips / ellipse pada primitif/basic.py

Lesson Learnt (Code, Print Screen Hasil Karya, dan Komentar)

1. Lingkaran
Pada main.py
`from primitif.basic import circleMidpoint`
`def draw():`
`  #lingkaran`
`  circleMidpoint(py5.width / 2, py5.height / 2, 100)`
Pada basic.py

```python
def circleMidpoint(xCenter, yCenter, radius):
    x = 0
    y = radius
    p = 1 - radius

    circlePlotPoints(xCenter, yCenter, x, y)

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

def circlePlotPoints(xCenter, yCenter, x, y):
    py5.point(xCenter + x, yCenter + y)
    py5.point(xCenter - x, yCenter + y)
    py5.point(xCenter + x, yCenter - y)
    py5.point(xCenter - x, yCenter - y)
    py5.point(xCenter + y, yCenter + x)
    py5.point(xCenter - y, yCenter + x)
    py5.point(xCenter + y, yCenter - x)
    py5.point(xCenter - y, yCenter - x)
```



Komentar:
Dari folder bernama “primitive” ada file bernama ”basic”, di dalam file ada modul yang bernama circleMidpoint
circleMidpoint punya 3 parameter: xCenter dan yCenter = koordinat pusat lingkaran, dan radius.
circlePlotPoints akan menggambar 8 titik sekaligus. Perlu 8 titik karena dari algoritma midpoint circle. Dengan menggambar 8 titik secara simetris, hanya perlu menghitung satu sektor dari lingkaran kemudian dapat mencerminkan hasilnya ke sektor-sektor lainnya.

2. Ellipse

Pada main.py

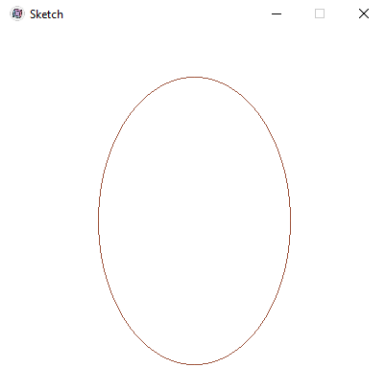
```python
from primitif.basic import ellipse_midpoint
```

```python
def draw():
    #lingkaran
    #circleMidpoint(py5.width / 2, py5.height / 2, 100)

    #ellipse
    ellipse_midpoint(py5.width / 2, py5.height / 2, 100, 150)
    py5.stroke(randint(0,255),randint(0,255),randint(0,255),255)
```

basic.py

```python
def ellipse_midpoint(xCenter, yCenter, Rx, Ry):
    def ellipse_plot_points(xCenter, yCenter, x, y):
        py5.point(xCenter + x, yCenter + y)
        py5.point(xCenter - x, yCenter + y)
        py5.point(xCenter + x, yCenter - y)
        py5.point(xCenter - x, yCenter - y)
    Rx2 = Rx * Rx
    Ry2 = Ry * Ry
    twoRx2 = 2 * Rx2
    twoRy2 = 2 * Ry2
    p = 0
    x = 0
    y = Ry
    px = 0
    py = twoRx2 * y

    # Plot the first set of points
    ellipse_plot_points(xCenter, yCenter, x, y)

    # Region 1
    p = round(Ry2 - (Rx2 * Ry) + (0.25 * Rx2))
```

```python
    # Region 1
    p = round(Ry2 - (Rx2 * Ry) + (0.25 * Rx2))
    while px < py:
        x += 1
        px += twoRy2
        if p < 0:
            p += Ry2 + px
        else:
            y -= 1
            py -= twoRx2
            p += Ry2 + px - py
        ellipse_plot_points(xCenter, yCenter, x, y)

    # Region 2
    p = round(Ry2 * (x + 0.5) ** 2 + Rx2 * (y - 1) ** 2 - Rx2 * Ry2)
    while y > 0:
        y -= 1
        py -= twoRx2
        if p > 0:
            p += Rx2 - py
        else:
            x += 1
            px += twoRy2
            p += Rx2 - py + px
        ellipse_plot_points(xCenter, yCenter, x, y)

```



Komentar:
Menggambar titik titik ellipse dengan fungsi ellipse_plot_points
Region 1 untuk menggambar setengah ellipse untuk menghitung titik-titik pada region pertama dari ellipse (kuadran atas) dan mencerminkannya ke kuadran lain.
Region 2 untuk menggambar sisa setengah ellipse untuk menghitung titik titik pada region kedua dari ellipse (kuadran atas bawah) dan mencerminkannya ke kuadran lain.
Panjang sumbu horizontal 100, dan panjang sumbu vertikal 150.

Kedua program menggunakan implementasi line bresenham.

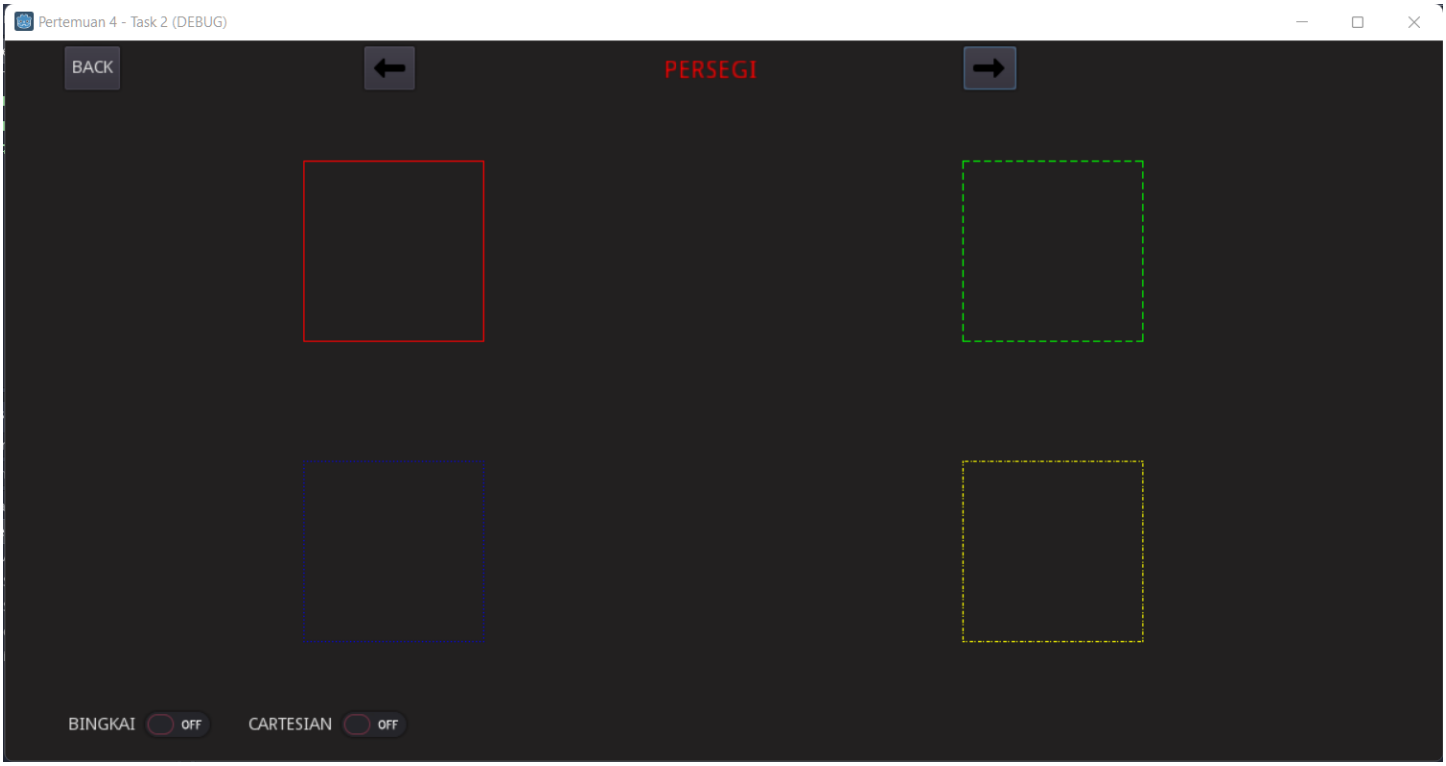
TASK 2: MEMBUAT FUNGSI BENTUK DASAR

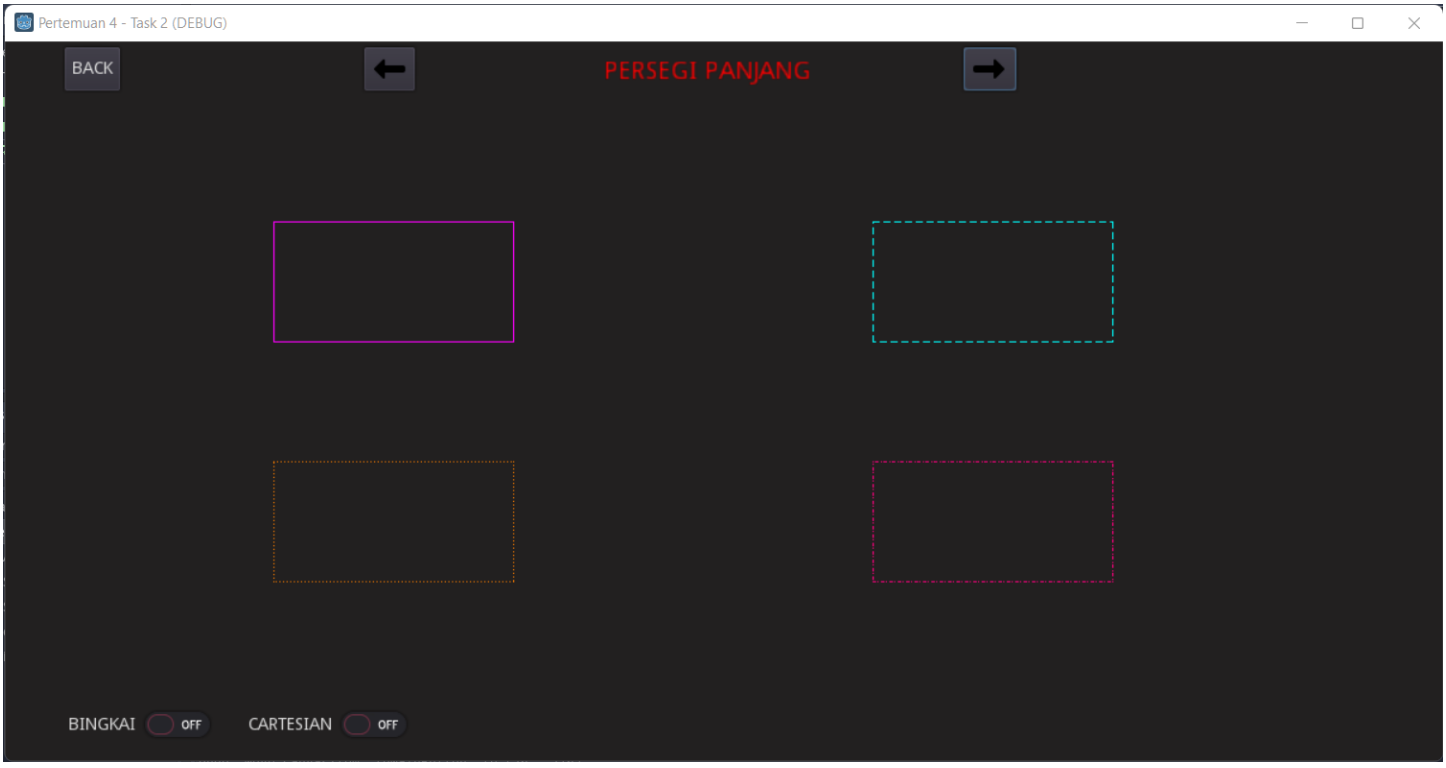
1. Buatlah Fungsi-fungsi Bentuk Dasar menggunakan algoritma generalisasi line bersenham sbb: Persegi, Persegi Panjang, Segitiga Siku-Siku, dan Trapesium Siku-Siku, Lingkaran dan Ellips
2. Posisikan Bentuk Dasar menjadi 4 Quadran.

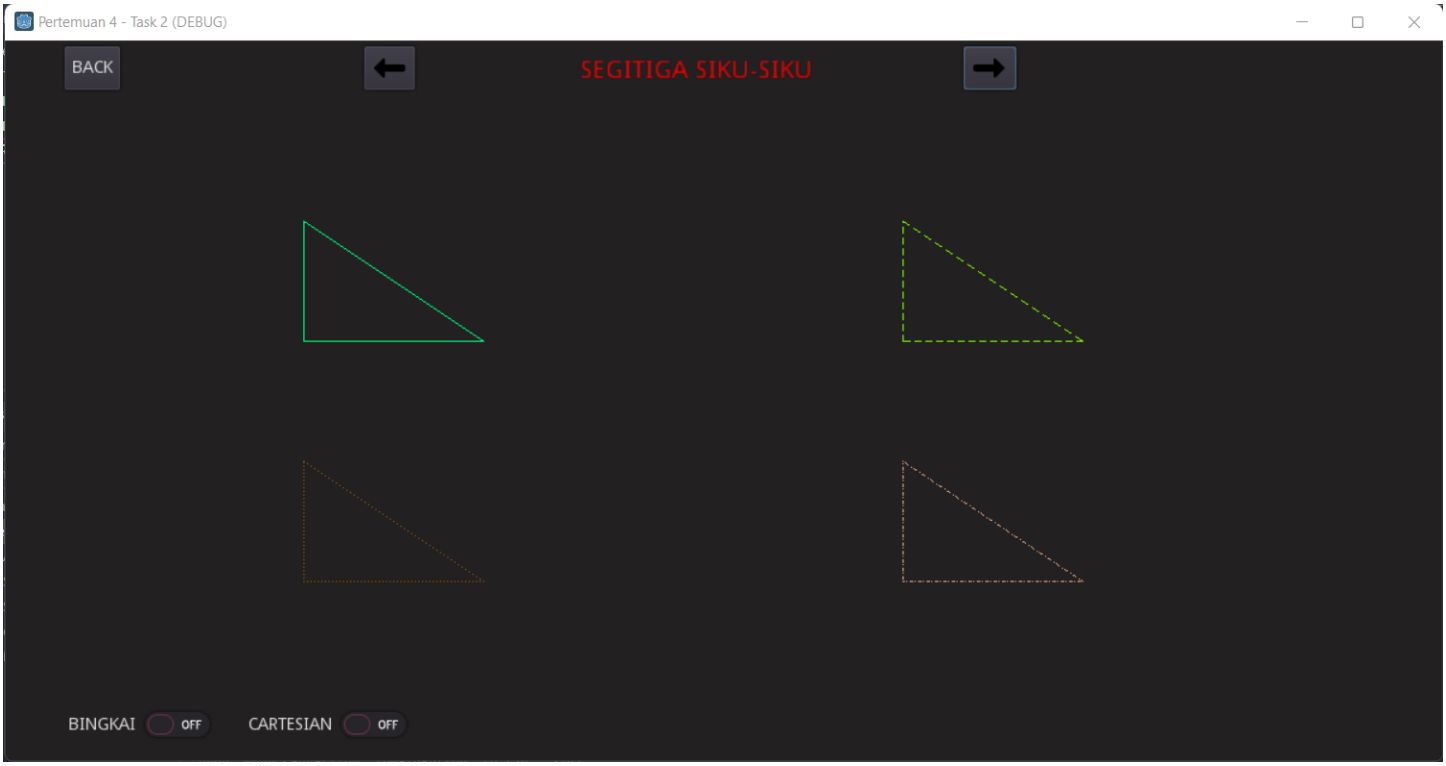
Untuk mengubah posisi dapat menggunakan fungsi convert to cartesian berikut

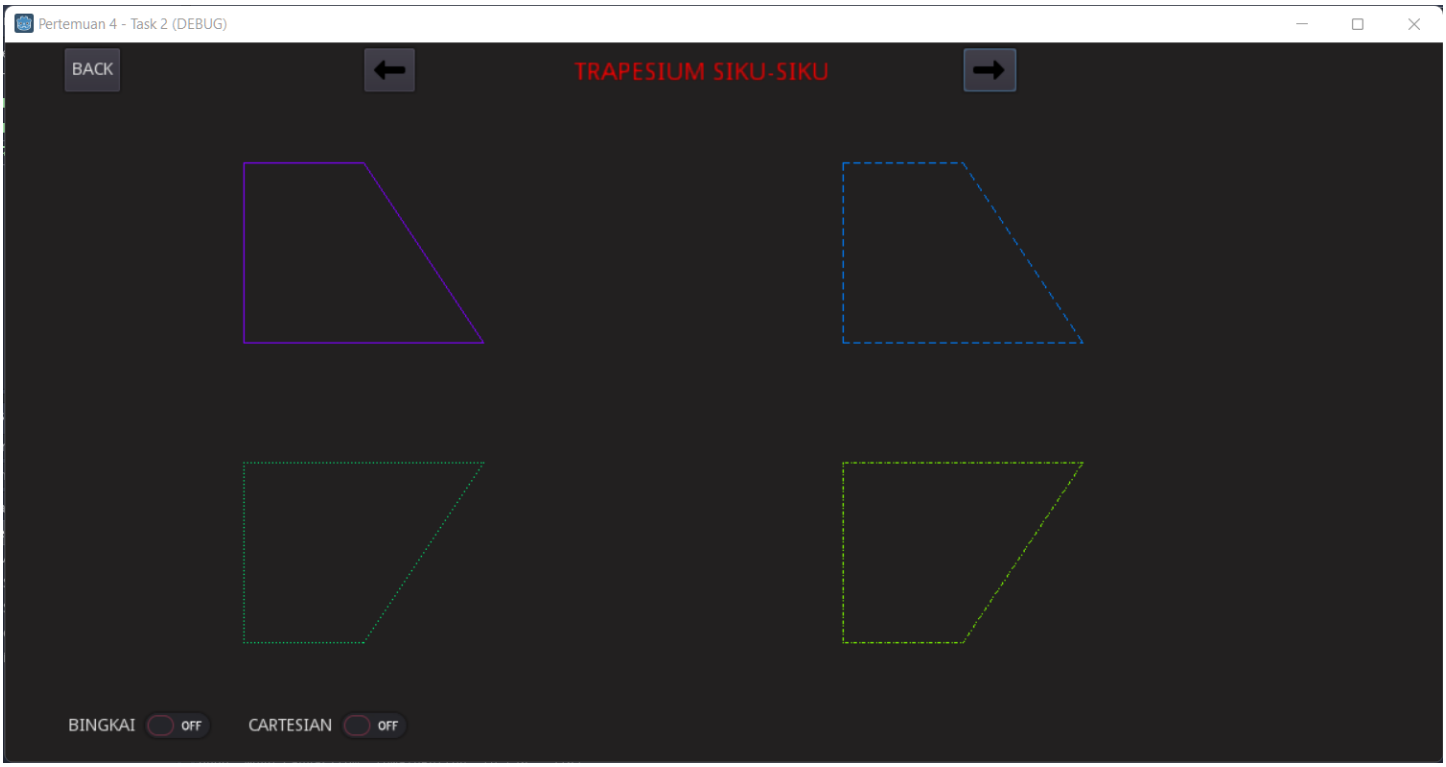
Utility.py

```
import math

def convert_to_pixel(xa, ya, xb, yb, width, height, margin):
    return [margin+xa, height-margin-ya, margin+xb, height-margin-yb]

def convert_to_cartesian(xa, ya, xb, yb, width, height, margin):
    axis = math.ceil(width/2)
    ordinat = math.ceil(height/2)
    return [axis+xa, ordinat-ya, axis+xb, ordinat-yb]
```


Pertemuan 4 - Task 2 (DEBUG)
BACK
PERSEGI
BINGKAI OFF
CARTESIAN OFF


Pertemuan 4 - Task 2 (DEBUG)
BACK
PERSEGI PANJANG
BINGKAI OFF
CARTESIAN OFF



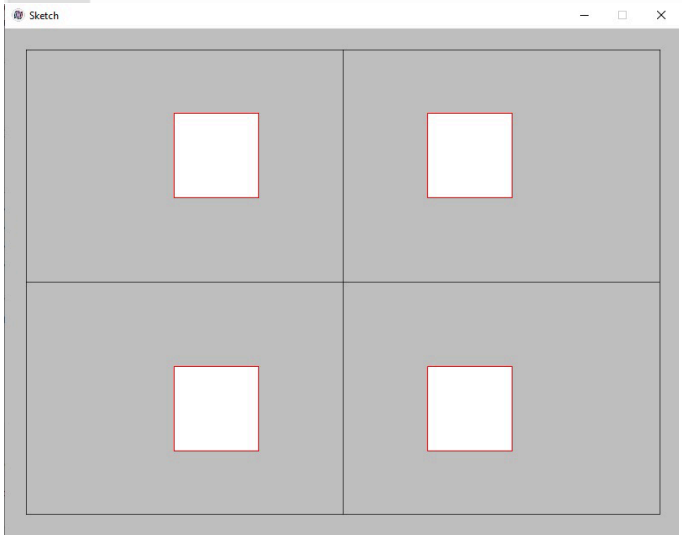


Lesson Learnt (Code, Print Screen Hasil Karya, dan Komentar)
1. Persegi Main.py

```
    #persegi
    # Ukuran dan posisi persegi
    sisi = 100
    x, y = 100, 100  #Koordinat yang diinginkan

    # Convert x and y to cartesian coordinates
    cartesian_coords = primitif.utility.convert_to_cartesian(x, y, py5.width, py5.height, 25)
    #kuadran 1
    primitif.basic.persegi(cartesian_coords[0], cartesian_coords[1], sisi, c=[255, 0, 0, 255])
    #kuadran 2
    primitif.basic.persegi(cartesian_coords[1], cartesian_coords[1], sisi, c=[255, 0, 0, 255])
    #kuadran 3
    primitif.basic.persegi(cartesian_coords[1], cartesian_coords[0], sisi, c=[255, 0, 0, 255])
    #kuadran 4
    primitif.basic.persegi(cartesian_coords[0], cartesian_coords[0], sisi, c=[255, 0, 0, 255])
```

Basic.py

```
def persegi(xa, ya, sisi, c=[0, 0, 0, 255]):
    py5.stroke(c[0], c[1], c[2], c[3])
    py5.begin_shape()
    py5.vertex(xa, ya)
    py5.vertex(xa + sisi, ya)
    py5.vertex(xa + sisi, ya - sisi)
    py5.vertex(xa, ya - sisi)
    py5.end_shape(py5.CLOSE)
```



Komentar: Pada modul def persegi di basic.py setiap sisi dari perseginya digambar. Parameter yang dibutuhkannya sebanyak 4

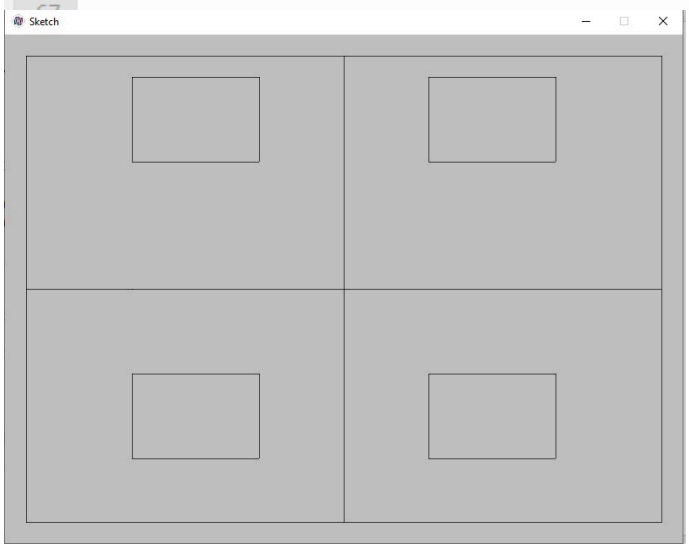
2. Persegi Panjang

Main.py

```
    #persegi panjang
    # Ukuran dan posisi persegi panjang
    panjang = 150
    lebar = 100
    x, y = 100, 150  #koordinat yang diinginkan
    # Convert x and y ke koordinat kartesian
    cartesian_coords = primitif.utility.convert_to_cartesian(x, y, py5.width, py5.height, 25)
    #Kuadran 1
    primitif.basic.persegi_panjang(cartesian_coords[0], cartesian_coords[1], panjang, lebar)
    #Kuadran 2
    primitif.basic.persegi_panjang(cartesian_coords[1], cartesian_coords[1], panjang, lebar)
    #Kuadran 3
    primitif.basic.persegi_panjang(cartesian_coords[1], cartesian_coords[0], panjang, lebar)
    #Kuadran 4
    primitif.basic.persegi_panjang(cartesian_coords[0], cartesian_coords[0], panjang, lebar)
```

basic.py

```python
def persegi_panjang(xa, ya, panjang, lebar):
    c=[0,0,0,255]
    # Hitung koordinat akhir (xb, yb) berdasarkan panjang dan lebar
    xb = xa + panjang
    yb = ya - lebar

    # Gambar sisi atas
    py5.points(primitif.line.line_bresenham(xa, ya, xb, ya))

    # Gambar sisi kanan
    py5.points(primitif.line.line_bresenham(xb, ya, xb, yb))

    # Gambar sisi bawah
    py5.points(primitif.line.line_bresenham(xb, yb, xa, yb))

    # Gambar sisi kiri
    py5.points(primitif.line.line_bresenham(xa, yb, xa, ya))

    pass
```



Komentar: pada modul def persegi_panjang pada basic.py juga semua sisi digambar. Di sini mengalami error bahwa modul line_bresenham hanya memiliki 4 parameter, tetapi yang dinyatakan di modul persegi_panjang ada 5. Oleh karena itu, saya menghapus 1 parameter, yaitu warna. Program berhasil dijalankan

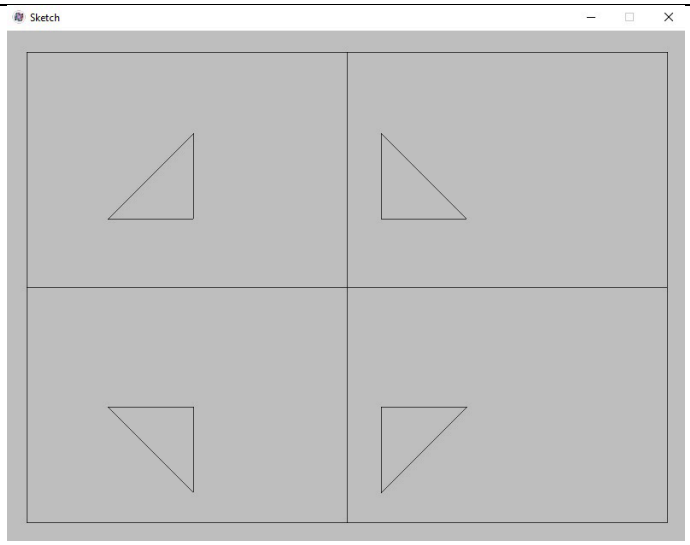
3. Segitiga Siku-Siku

Main.py

```python
    # Convert x dan y ke koordinat kartesian
    cartesian_coords = primitif.utility.convert_to_cartesian(x, y, py5.width, py5.height, 25)

    # Gambar segitiga siku kuadran 1
    primitif.basic.segitiga_siku(cartesian_coords[0], cartesian_coords[1], 100, 100)
    # Gambar segitiga siku kuadran 2
    primitif.basic.segitiga_siku(cartesian_coords[1], cartesian_coords[1], -100, 100)
    # Gambar segitiga siku kuadran 3
    primitif.basic.segitiga_siku(cartesian_coords[1], cartesian_coords[0], -100, -100)
    # Gambar segitiga siku kuadran 4
    primitif.basic.segitiga_siku(cartesian_coords[0], cartesian_coords[0], 100, -100)
```

Basic.py

```python
def segitiga_siku(xa, ya, alas, tinggi):

    py5.points(primitif.line.line_bresenham(xa, ya, xa + alas, ya))  # Sisi bawah
    py5.points(primitif.line.line_bresenham(xa, ya, xa, ya - tinggi))  # Sisi kiri
    py5.points(primitif.line.line_bresenham(xa, ya - tinggi, xa + alas, ya))  # Hypotenuse
    pass
```



Komentar:
pada modul def segitiga_siku pada basic.py juga semua sisi digambar. Agar tercipta pencerminan, digunakanlah rumus cermin terhadap sb-x dan sb-y.

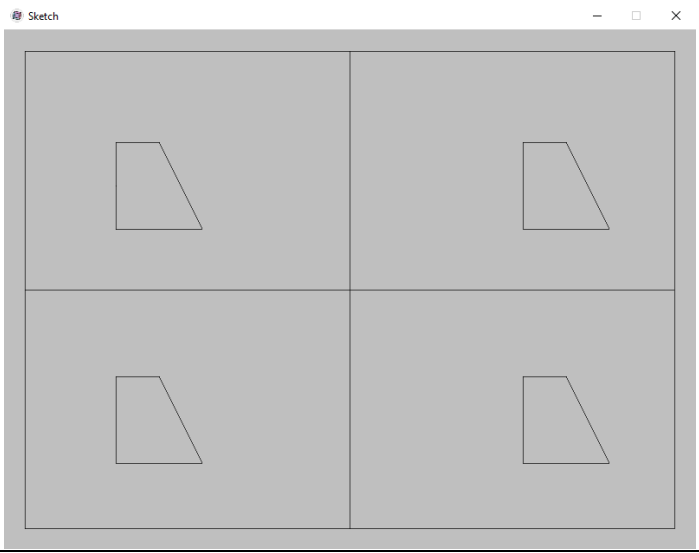
4. Trampesium Siku-Siku

Pada main.py

```
    #trampesium
a = [130, 600]
b = [130, 400]
for x in a:
    for y in b:
        primitif.basic.trapesium_siku(x, y, 50, 100, 100, c=[0,0,0,255])
```

basic.py

```
75  def trapesium_siku(xa, ya, aa, ab, tinggi, c=[0,0,0,255]):
76      xb = xa + aa
77      xc = xa + ab
78      yb = ya + tinggi
79      py5.points(primitif.line.line_bresenham(xa, ya, xb, ya))
80      py5.points(primitif.line.line_bresenham(xa, yb, xc, yb))
81      py5.points(primitif.line.line_bresenham(xc, yb, xb, ya))
82      py5.points(primitif.line.line_bresenham(xa, ya, xa, yb))
83      pass
```



Lesson learn:
Program untuk membuat bangun datar. Modul modul dibuat di basic.py lalu dipanggil di main.py. setiap memanggil, harus menjadikan comment panggilan bangun datar yang lainnya. Error yang paling sering ditemukan adalah unexpected indent yaitu adanya tab, tab harus dihapus. Primitif.Pertemuan2 juga error, ganti menjadi primitif.task3, karena ada file bernama task3 di folder primitif. Yang bagus adalah menggunakan cartesius convert agar bangun datar bisa langsung dipetakkan di kuadran yang diinginkan, jika ingin pencerminan, bisa lihat rumus pencerminan. Trampesium belum berhasil menggunakan cartesius convert. Dengan menggunakan modul draw_grid, draw_margin, dan draw_kartesian di basic.py langsung tergambar meskipun hasil dari main.py tidak muncul selama tidak ada error

TASK 4: MEMBUAT KARYA 2D TIC TAC TOE

Buatlah board tic tac toe 3x3 dengan syarat sbb

1. Pemain 1 berupa X dan Pemain 2 Berupa Lingkaran
2. Letakan Pemain dengan Kondisi Pemain 1 Menang
3. Letakan Pemain dengan Kondisi Tidak ada Pemenang

Lesson Learnt (Code, Print Screen Hasil Karya, dan Komentar)

Membuat kotak 3x3

Main.py

```
8  def setup():
9      py5.size(800, 600)
10     py5.rect_mode(py5.CENTER)
11     py5.background(191)
12     primitif.basic.draw_margin(py5.width, py5.height, 30, c=[0,0,0,255])
13
14     py5.points(primitif.line.line_bresenham(py5.width*33/100, 30, py5.width*33/100, py5.height-30))
15     py5.points(primitif.line.line_bresenham(py5.width*66/100, 30, py5.width*66/100, py5.height-30))
16     py5.points(primitif.line.line_bresenham(30, py5.height*33/100, py5.width-30, py5.height*33/100))
17     py5.points(primitif.line.line_bresenham(30, py5.height*66/100, py5.width-30, py5.height*66/100))
```

Pemain 1 menang

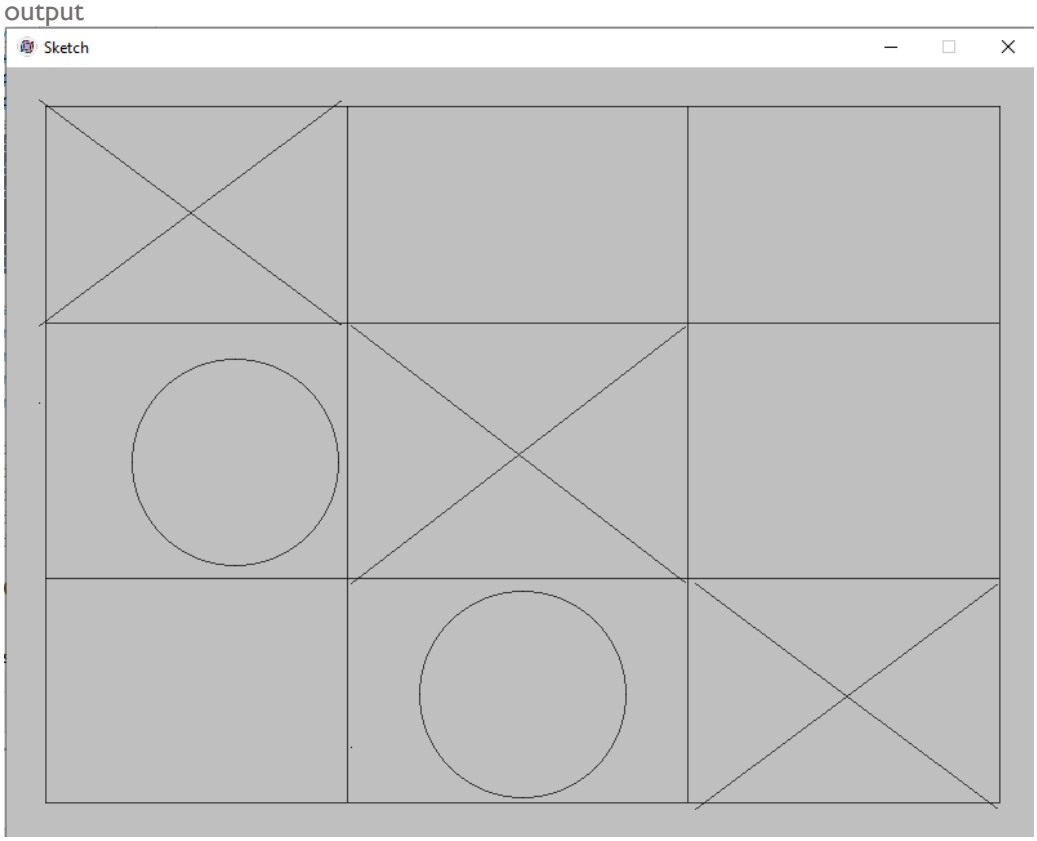
Basic.py

```
47  def kali(xa, ya, panjang, c=[0,0,0,255]):
48      xb = xa + panjang
49      yb = ya + panjang
50      py5.points(primitif.line.line_bresenham(xa, ya, xb+60, yb))
51      py5.points(primitif.line.line_bresenham(xb+60, ya, xa, yb))
52
```

Main.py

```
19      primitif.basic.kali(25, 25, 175, c=[0,0,0,255])
20      primitif.basic.kali(267, 200, 200, c=[0,0,0,255])
21      primitif.basic.kali(534, 400, 175, c=[0,0,0,255])
22
23      circleMidpoint(py5.width / 2, 30 + py5.height * 38 / 50, 80)
24      circleMidpoint(py5.width / 4.5, 90 + py5.height * 18 / 50, 80)
```

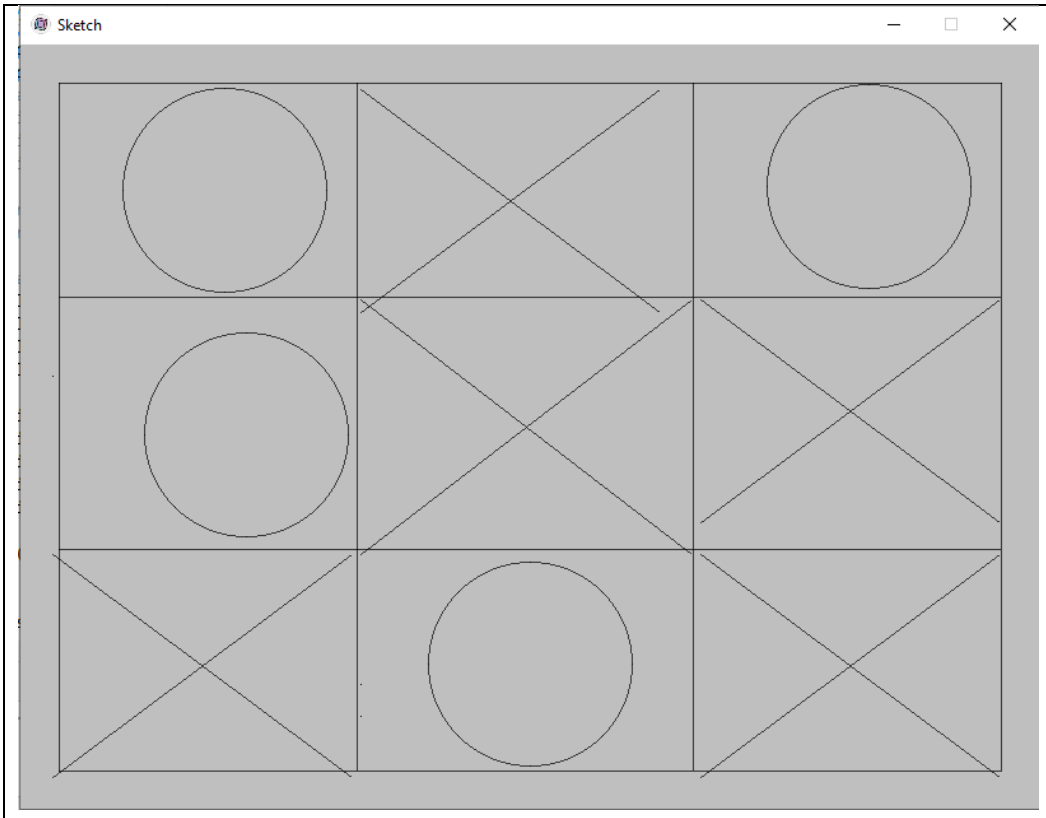
output



Tidak ada pemain yang menang
Basic.py sama seperti praktikum 0, menggunakan fungsi midpoint circle

Main.py

```
26      #tidak ada pemenang
27      circleMidpoint(py5.width / 2, 30 + py5.height * 38 / 50, 80)
28      circleMidpoint(py5.width / 1.2, 15 + py5.height * 8 / 50, 80)
29      circleMidpoint(py5.width / 4.5, 90 + py5.height * 18 / 50, 80)
30      circleMidpoint(py5.width / 5, 90 + py5.height * 2 / 50, 80)
31
32      primitif.basic.kali(25, 400, 175, c=[0,0,0,255])
33      primitif.basic.kali(534, 200, 175, c=[0,0,0,255])
34      primitif.basic.kali(267, 200, 200, c=[0,0,0,255])
35      primitif.basic.kali(534, 400, 175, c=[0,0,0,255])
36      primitif.basic.kali(267, 35, 175, c=[0,0,0,255])
37
```



Lesson learned:
Untuk pemetaan papan 3x3, berarti perlu ada 4 garis. Saya membuat 2 garis horizontal dan 2 garis vertikal dengan menghitung dari grid yang dibagi menjadi 3 bagian. Saya mendapatkan referensi ukuran.

Untuk ketika pemain 1 menang, perlu ada 3 X dan 2 O karena pemain 1 adalah X, jadi jumlah X akan lebih banyak. Untuk membuat X, saya meperkirakan koordinat dari setiap X, beberapa kali percobaan hingga mendapatkan pemetaan yang dirasa tepat. Untuk membuat O saya mengambil modul lingkaran pada basic.py task 0, lalu melakukan beberapa kali pemetaan untuk menghasilkan hasil yang dirasa tepat.

Untuk ketika tidak ada pemain yang menang, saya memodifikasi dari kode pemain 1 menang lalu mengubah sedikit koordinatnya

Hal yang harus dilakukan untuk meningkatkan pemahaman:
1. Mencoba mengotret dengan pensil dan kertas
2. Sering mencoba agar terbiasa dengan fungsi fungsi di py5

PENGUMPULAN

Ikuti Format yang diberikan di Google Classroom.