MODUL IV

KOMPUTER GRAFIK 2D
ATRIBUTE GRAFIK DAN TRANSFORMASI 2D I & 2

D3 TEKNIK INFORMATIKA
JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA
POLITEKNIK NEGERI BANDUNG

MAHASISWA 029 | KOMPUTER GRAFIK | SEPTEMBER, 14 2023

CONTENTS

TIC TAC TOE GAME

https://playtictactoe.org/

Permainan Tic-Tac-Toe atau catur jawa atau XOX merupakan permainan classic yang digunakan untuk belajar pemrograman game. Selain Tic-Tac-Toe ada minesweeper, digger, snake, dan pong. Aturan main Tic-Tac-Toe sangat sederhana, 2 pemain berusaha menyelesaikan kondisi kemenangan yaitu ketika terdapat tiga tanda yang sama pada posisi vertikal, horizontal atau diagonal secara berurutan. Sebaliknya permainan akan draw jika 2 pemain tidak dapat menghasilkan kondisi tersebut.

Pemain 1 menulis X dan Pemain 2 menulis O pada papan 3x3

Pada praktikum sebelumnya kita telah membuat bentuk dasar lingkaran, garis dan kali dan akan memanfaatkan kode tersebut untuk membuat permainan ini.

Fitur-fitur Permainan Tic Tac Toe
1. Papan Grid 3x3 2. Pemain 1 bisa menuliskan X di Papan & Pemain 2 bisa menuliskan O di Papan 3. Kolom yang sudah dituliskan tidak boleh dituliskan kembali 4. Kondisi draw tidak ada X atau O yang sesuai dengan kondisi (Vertikal, Horizontal, Diagonal) 5. Kondisi menang ada X atau O yang sesuai dengan kondisi (Vertikal, Horizontal, Diagonal)

PYTHON 201 : CLASS

Python 201.

1. Abstraksi
2.

KARYA 2D DENGAN OOP

Tictactoe

Asteroid

Stick Man

Kendaraan

ATTRIBUTE GRAFIK

Terdapat beberapa atribut Grafik yang dapat diimplemntasikan pada Komputer Grafik contohnya variasi dari bentuk dasar seperti Titik – titik, Titik – garis – titik, Garis – garis kosong garis garis, Dan variasi lainnya. Selain itu pengaturan dari tebal dan tipisnya bentuk 2D yang dihasilkan.

Secara implementasi untuk object 2D yang dibangun dari garis dapat diimplementasikan setelah Kumpulan points tercipta oleh algoritma line dda atau line bersenham, sedangkan pada lingkaran dan ellips pada algoritma Pembangunan lingkaran dan ellips atau dilakukan perubahan algoritma sehingga luaran dari algoritma lingkaran dan garis berupa array seperti pada garis.

TRANSFORMASI 2D

Transformasi 2D

TRANSLASI 2D

Translasi 2D

SCALING 2D

Scaling 2D

SHEAR 2D

Shear 2D

TASK PRAKTIKUM

TASK 1-3: REVIEW DAN EKSPLORASI

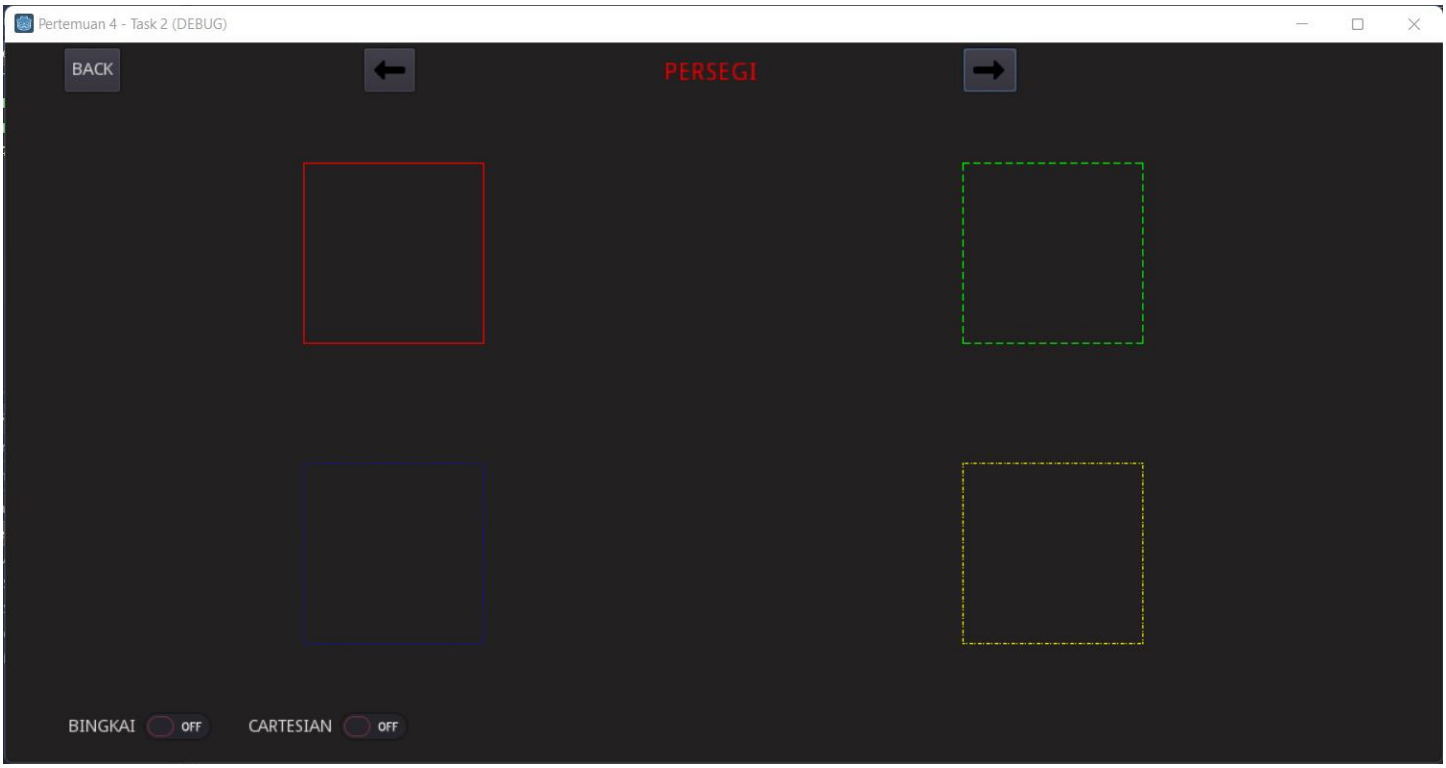
1. Download dan Buka File Pendukung Praktikum [KG2023_2X_001_D3_2022]_Modul4

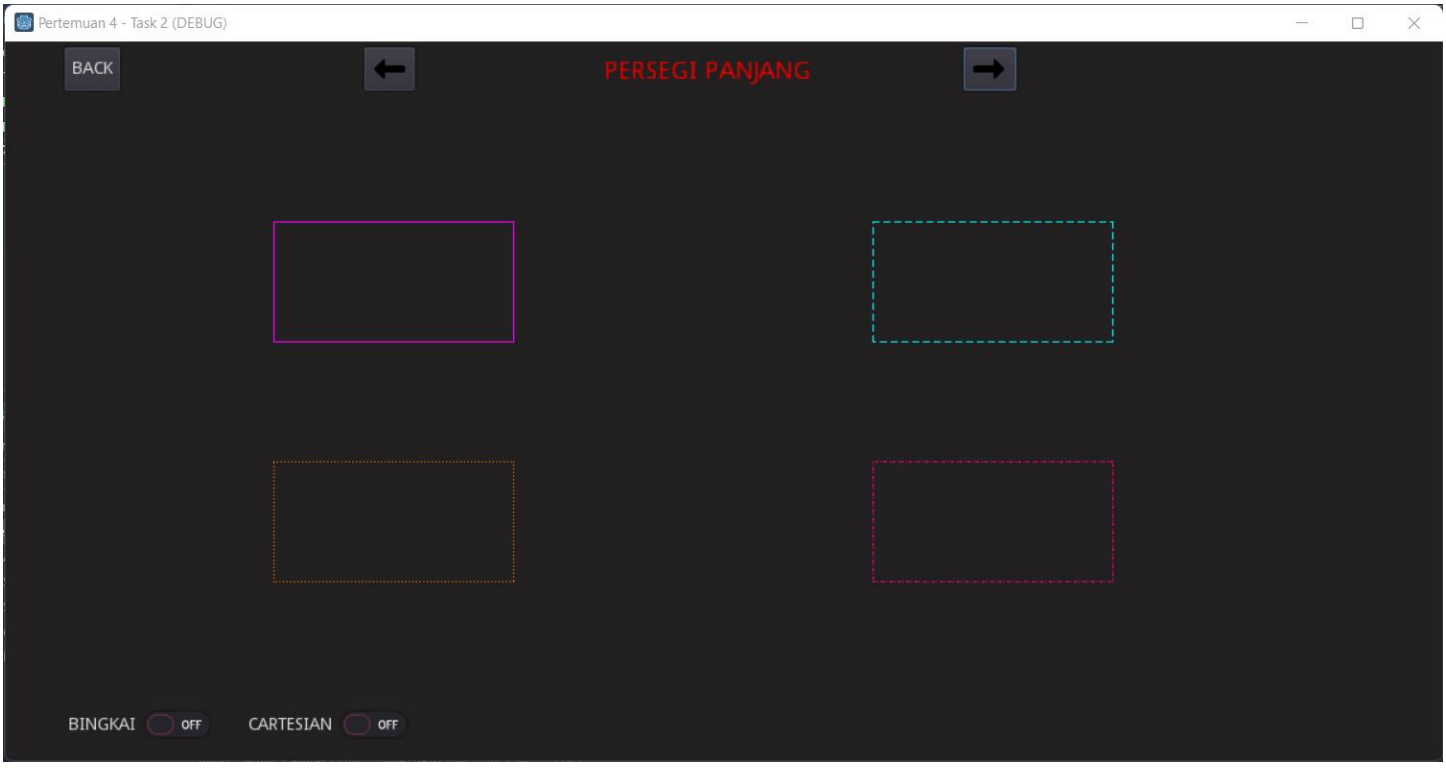
2. 001_Task01 adalah file tictactoe silakan coba dan analisa kode tersebut diskusikan jika ada yang tidak dipahami
3. 001_Task02 adalah file untuk memahami cara kerja object oriented di python
4. 001_Task03 adalah file untuk memahami cara kerja OOP terimplementasi pada tugas sebelumnya (Tictactoe, asteroid, stickman, dan kendaran).

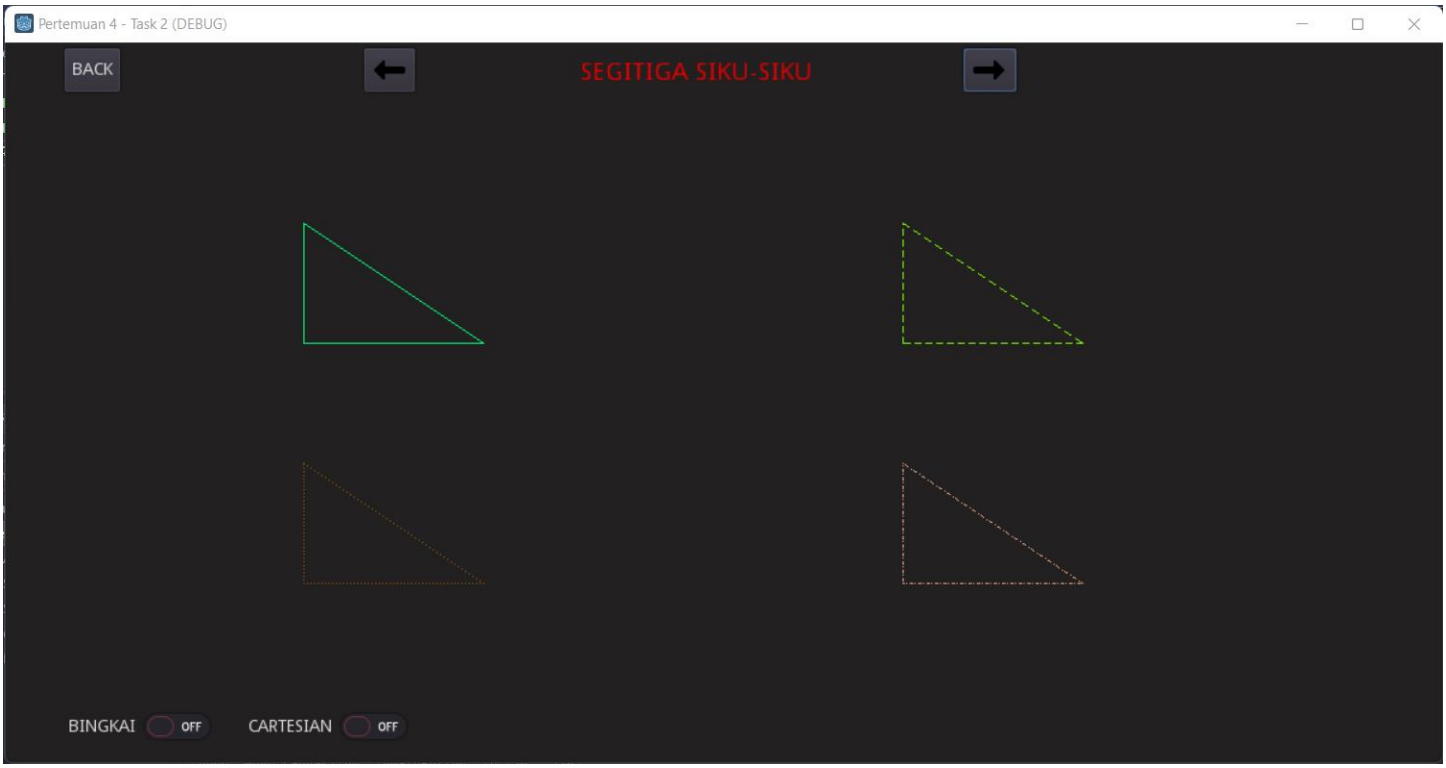
Lesson Learnt (Code, Print Screen Hasil Karya, dan Komentar)

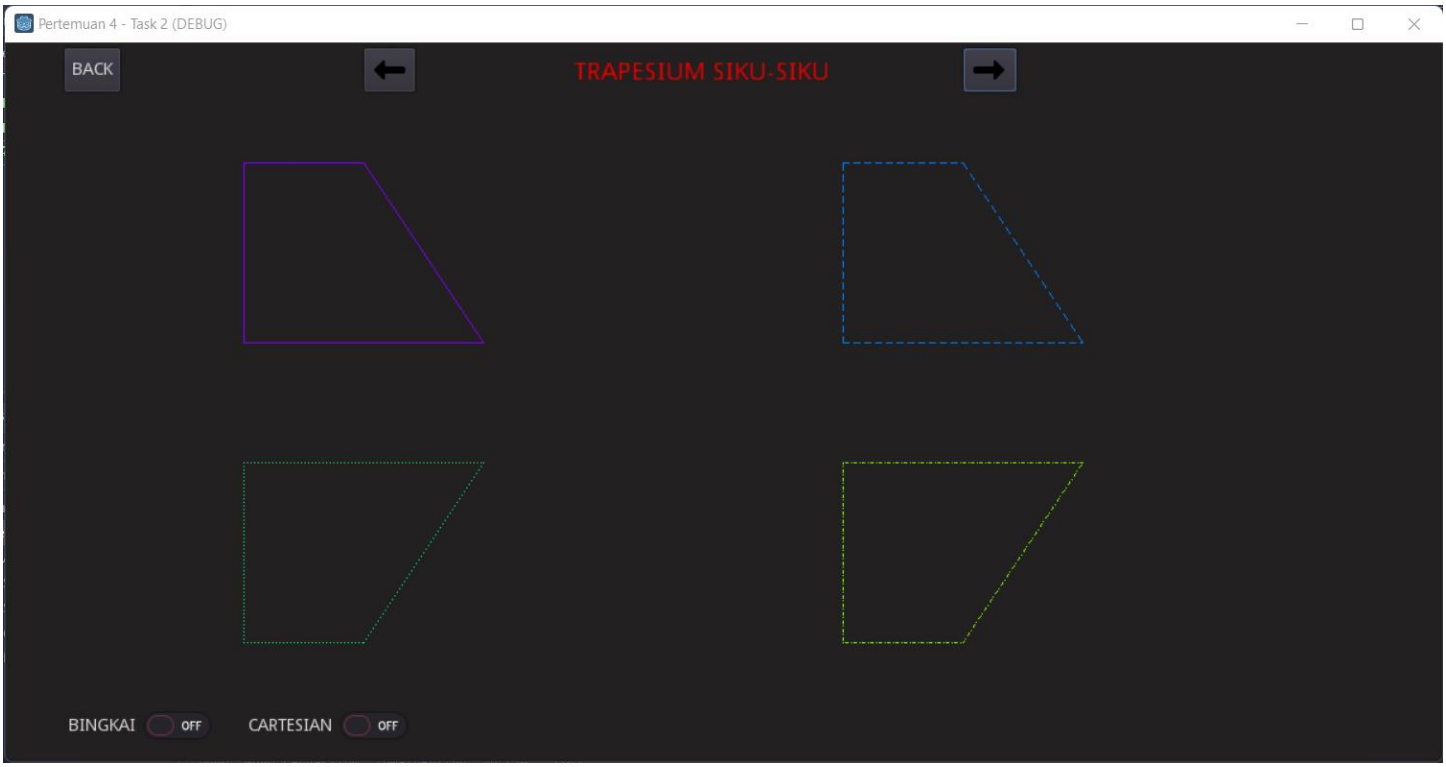
TASK 4: MEMBUAT FUNGSI BENTUK DASAR DENGAN ATTRIBUT GRAFIK

1. Copy Task 3
2. Buatlah Fungsi-fungsi Bentuk Dasar: Persegi, Persegi Panjang, Segitiga Siku-Siku, dan Trapesium Siku-Siku, Lingkaran dan Ellips dengan Attribut Grafik sbb:
   Titik – titik
   Titik – garis – titik
   Garis – garis kosong garis garis
   Dan variasi lainnya.
3. Konversi Karya 2D OOP dengan attribute grafik.


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


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


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


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

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

Basic.py
 	
`def draw_garis_gap(pts, jarak):`
`  py5.stroke(0, 0, 0, 255)`
`  for i in range(len(pts)):`
`  x, y = pts[i]`
`  if (i + 1) % jarak == 0:`
`  continue`
`  else:`
`  py5.point(x, y)`
 	
`def draw_garis_titik(pts, jarak):`
`  py5.stroke(0, 0, 0, 255)`
`  for i in range(len(pts)):`
`  x, y = pts[i]`
 	
`  if i % 4 == 0 and i > 0:`
`  x1, y1 = pts[i - 1]`
`  x_gap = (x + x1) // 2`
`  y_gap = (y + y1) // 2`
`  py5.point(x_gap, y_gap`
 	
`  if (i + 1) % jarak == 0:`
`  continue`
`  else:`
`  py5.point(x, y)`
	Fungsi ini menggambar garis antara titik dalam daftar pts dengan jarak tertentu. Setiap garis gambar antara titik yang berjarak satu sama lain, dan titik yang berjarak lebih jauh diabaikan.

Main.py
 	
`def draw():`
`  py5.background(255)`
`  time = 30`
`  koordinat = [(50, 100), (-150, 100), (-150, -50), (50, -50)]`
`  garis = ["garis", "titik", "garis-titik", "garis-kosong"]`
`  basic.draw_kartesian(py5.width, py5.height, 25)`
`  index_garis = 0`
	Untuk menempatkan bangun datar pada empat kuadran yang isinya adalah titik x dan y dari tiap kuadran, koordinat didefinisikan. Selanjutnya, untuk mengetahui garis mana yang akan digunakan untuk gambar bangun datar di masa mendatang, index_garis didefinisikan mulai dari 0.

```
    if config.anim <= time:#persegi
        for x, y in koordinat:
            koor_persegi = utility.convert_to_cartesian(x, y, py5.width, py5.height, 20)
            persegi = basic.persegi(koor_persegi[0], koor_persegi[1], 70)
            basic.draw_bentuk(persegi, garis[index_garis])
            index_garis += 1
            if index_garis >= len(garis):
                index_garis = 0
    elif config.anim <= 2*time:#persegi panjang
        for x, y in koordinat:
            koor_persegi_panjang = utility.convert_to_cartesian(x, y, py5.width, py5.height, 20)
            persegi_panjang = basic.persegi_panjang(koor_persegi_panjang[0], koor_persegi_panjang[1], 100, 50)
            basic.draw_bentuk(persegi_panjang, garis[index_garis])
            index_garis += 1
            if index_garis >= len(garis):
                index_garis = 0
    elif config.anim <= 3*time:#segitiga
        for x, y in koordinat:
            koor_segitiga = utility.convert_to_cartesian(x, y, py5.width, py5.height, 20)
            segitiga = basic.segitiga_siku(koor_segitiga[0], koor_segitiga[1], 100, 50)
            basic.draw_bentuk(segitiga, garis[index_garis])
            index_garis += 1
            if index_garis >= len(garis):
                index_garis = 0
    elif config.anim <= 4*time:#trapesium
        for x, y in koordinat:
            koor_trapesium = utility.convert_to_cartesian(x, y, py5.width, py5.height, 20)
            trapesium = basic.trapesium_siku(koor_trapesium[0], koor_trapesium[1], 50, 100, 60)
            basic.draw_bentuk(trapesium, garis[index_garis])
            index_garis += 1
            if index_garis >= len(garis):
                index_garis = 0
    elif config.anim <= 5*time:#lingkaran
        for x, y in koordinat:
            koor_lingkaran = utility.convert_to_cartesian(x, y, py5.width, py5.height, 20)
            lingkaran = basic.lingkaran (koor_lingkaran[0], koor_lingkaran[1], 40)
            basic.draw_bentuk(lingkaran, garis[index_garis])
            index_garis += 1
            if index_garis >= len(garis):
                index_garis = 0
    elif config.anim <= 6*time:#kali
        for x, y in koordinat:
            koor_kali = utility.convert_to_cartesian(x, y, py5.width, py5.height, 20)
            kali = basic.kali(koor_kali[0], koor_kali[1], 70)
            basic.draw_bentuk(kali, garis[index_garis])
            index_garis += 1
            if index_garis >= len(garis):
                index_garis = 0
    elif config.anim <= 7*time:#ellips
        for x, y in koordinat:
            koor_ellips = utility.convert_to_cartesian(x, y, py5.width, py5.height, 20)
            ellips = basic.ellips(koor_ellips[0], koor_ellips[1], 40,50)
            basic.draw_bentuk(ellips, garis[index_garis])
            index_garis += 1
            if index_garis >= len(garis):
                index_garis = 0
    if config.anim > 7*time:
        config.anim = 0
```

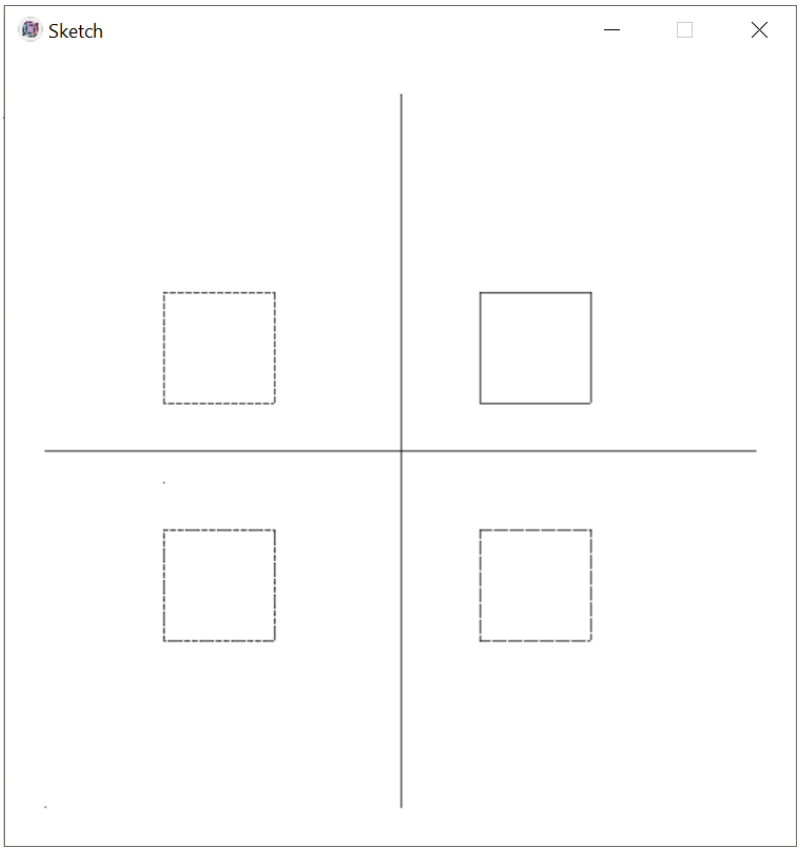
Untuk menempatkan bangun datar pada empat kuadran yang isinya adalah titik x dan y dari tiap kuadran, koordinat didefinisikan, kemudian, untuk mengetahui garis mana yang akan digunakan untuk gambar bangun datar nantinya, kemudian index_garis didefinisikan mulai dari 0.

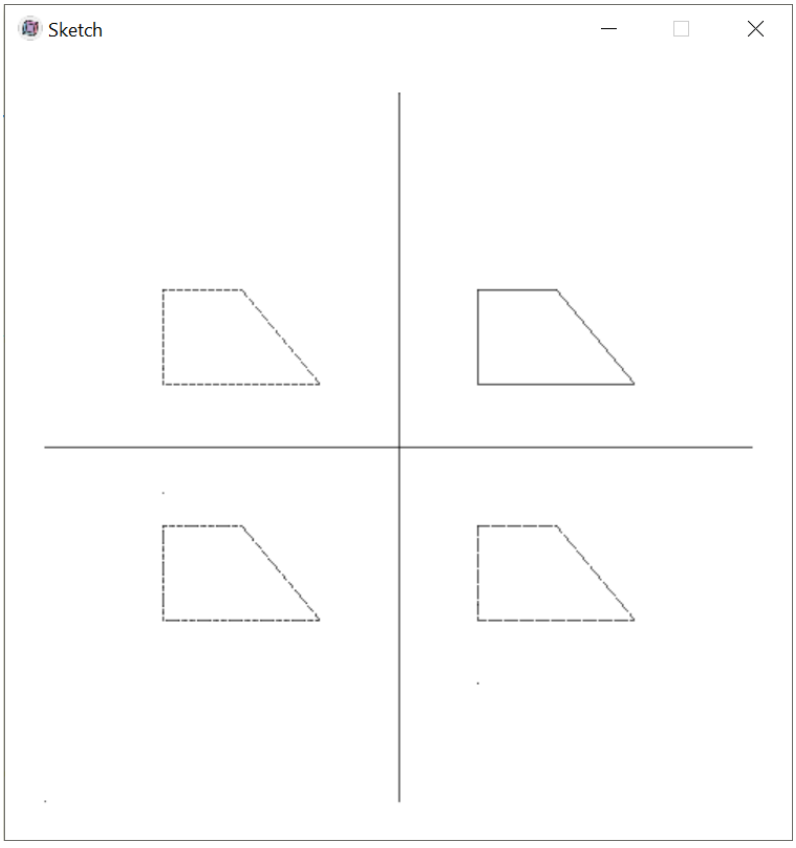
```
if config.anim > 7*time:
    config.anim = 0

config.anim += 1
```

Pada “anim” akan membuat looping sampai 7 bangun datar tellah terbuat, lalu jika sudah sampai 7 maka “anim” akan memulai dari awal atau dari 0.

CONTOH OUTPUT





TASK 6 MEMBUAT FUNGSI-FUNGSI TRANSFORMASI 2D,

1. Copy Task 5, buatlah fungsi-fungsi transformasi 2D dengan operasi matrix yang dibantu dengan numpy dengan menggunakan homogeneous coordinate dengan titik (0,0) adalah origin monitor.
2. Buatlah Translasi, Scaling, Shear, dan Rotasi

<table>
<tr><th colspan="2">Lesson Learnt (Code, Print Screen Hasil Karya, dan Komentar)</th></tr>
<tr><th>Print screen hasil karya</th><th>Komentar</th></tr>
<tr><td>Tranformasiv2.py


```
def rotate2D(angle, tx, ty):
    angle_rad = np.radians(angle)
    cos_theta = np.cos(angle_rad)
    sin_theta = np.sin(angle_rad)
    return np.array([
        [cos_theta, -sin_theta, tx],
        [sin_theta, cos_theta, ty],
        [0, 0, 1]
    ])

def scale2D(sx, sy, tx, ty, tm=None):
    return np.array([
        [sx, 0, 0],
        [0, sy, 0],
        [0, 0, 1]
    ])

def translate2D(tx, ty, tm = None):
    return np.array([
        [1, 0, tx],
        [0, 1, ty],
        [0, 0, 1]
    ])

def matrix_multiply(mat1, mat2):

    result = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
    for i in range(3):
        for j in range(3):
            for k in range(3):
                result[i][j] += mat1[i][k] * mat2[k][j]
    return result

def transformPoints2D(points, transformation_matrix):
    transformed_points = []
    for point in points:
        homogeneous_point = np.array([point[0], point[1], 1])
        transformed_point = np.dot(transformation_matrix, homogeneous_point)
        transformed_points.append(transformed_point[:2])  # Extract x and y
    return transformed_points
```

```
def shear_horizontal(points, shx):
    shear_matrix = shear2D_horizontal(shx)
    transformed_points = []
    for point in points:
        homogeneous_point = np.array([point[0], point[1], 1])
        transformed_point = np.dot(shear_matrix, homogeneous_point)
        transformed_points.append(transformed_point[:2])
    return transformed_points

def shear2D_vertical(shy, tm=None):
    return np.array([
        [1, 0, 0],
        [shy, 1, 0],
        [0, 0, 1]
    ])

def shear2D_horizontal(shx, tm=None):
    return np.array([
        [1, shx, 0],
        [0, 1, 0],
        [0, 0, 1]
    ])
```

</td><td>fungsi-fungsi ini adalah untuk melakukan serangkaian transformasi geometri pada titik-titik 2D, seperti rotate, scaling, Shear dan translasi. Output dari program ini akan berupa titik-titik yang telah diubah sesuai dengan transformasi yang telah didefinisikan menggunakan matriks transformasi yang sesuai.</td></tr>
<tr><td>Main.py</td><td>Ini adalah bentuk pemanggilan persegi untuk mengetahui apakah transformasi persegi</td></tr>
</table>

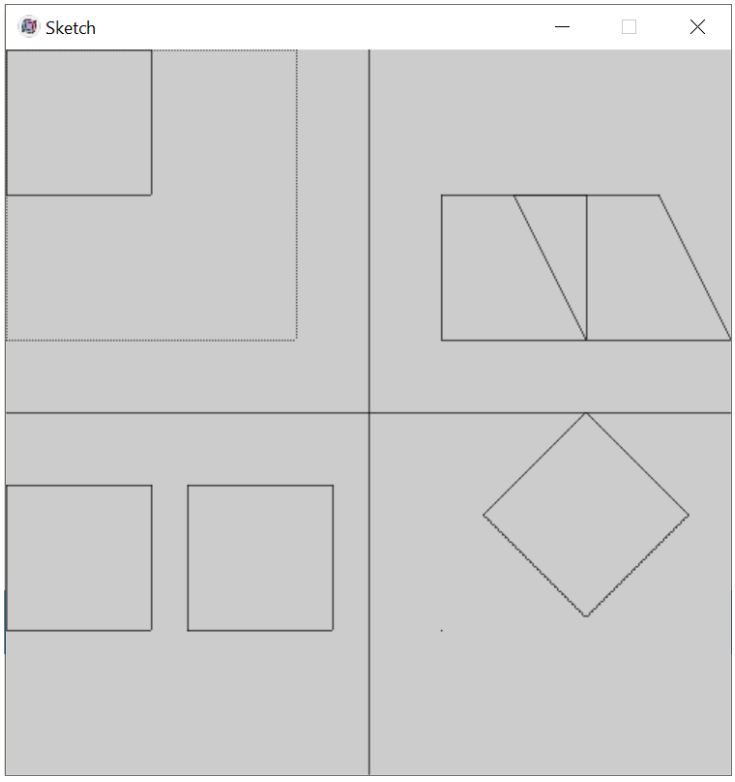
```
def setup():
    py5.size(500,500)
    primitif.basic.draw_kartesian(py5.width, py5.height, 0)
    persegi = primitif.basic.persegi(0,0,100)
    persegi1 = primitif.basic.persegi(300,100,100)
    persegi2 = primitif.basic.persegi(0,300,100)
    primitif.basic.draw_bentuk(persegi)
    primitif.basic.draw_bentuk(persegi1)
    primitif.basic.draw_bentuk(persegi2)
    tm = primitif.transformasiv2.rotate2D(45, 400, 250)
    print(tm)
    persegi4 = primitif.transformasiv2.transformPoints2D(persegi,tm)
    primitif.basic.draw_bentuk(persegi4)
    tm = primitif.transformasiv2.scale2D(2, 2, 400, 0)
    print(tm)
    persegi3 = primitif.transformasiv2.transformPoints2D(persegi,tm)
    primitif.basic.draw_bentuk(persegi3)
    tm = primitif.transformasiv2.translate2D(125,300)
    print(tm)
    persegi2 = primitif.transformasiv2.transformPoints2D(persegi,tm)
    primitif.basic.draw_bentuk(persegi2)

    shx = 0.5  # Shearing factor
    sheared_square = []
    for point in persegi1:
        x_sheared = point[0] + shx * point[1]
        y_sheared = point[1]
        sheared_square.append([x_sheared, y_sheared])

    primitif.basic.draw_bentuk(sheared_square)
```

tersebut berhasil atau tidak. Saya membuat tiga persegi biasa yang nantinya akan memiliki bentuk transformasi lain dari kuadran tersebut.

OUTPUT



Lesson Learnt
Beberapa transformasi dilakukan selama praktikum yang dilakukan: a. Translasi, atau pergeseran, adalah memindahkan semua titik pada suatu objek dengan jarak dan arah tertentu. b. Skala, atau skala, adalah memperbesar atau memperkecil suatu objek. c. Rotasi, atau perputaran, adalah memutar suatu objek dengan sumbu tetap. Dan, d. Shear, adalah transformasi yang dilakukan dengan "membebani" objek pada arah tertentu. • Matriks 3 kali 3 digunakan untuk transformasi 2D karena memungkinkan transformasi yang lebih kompleks dan memungkinkan translasi (pergeseran) ke dalam matriks, yang penting untuk mengubah posisi objek. Matriks 3 kali 3 juga memungkinkan penggabungan transformasi seperti rotasi, penskalaan, dan pergeseran dalam satu langkah, sehingga memudahkan perhitungan transformasi yang kompleks. • TransformPoints2D adalah fungsi yang mengubah matriks yang menunjukkan titik-titik dua dimensi yang ada menjadi titik-titik baru yang telah mengalami transformasi sesuai dengan matriks transformasi yang diberikan. Fungsi ini mengambil matriks yang mewakili sekumpulan titik, dan kemudian menerapkan transformasi menggunakan matriks transformasi yang diberikan pada setiap titik dalam matriks tersebut.

PENGUMPULAN

Ikuti Format yang diberikan di Google Classroom.