

Tugas Besar II

IF2123 Aljabar Geometri

“Simulasi Transformasi Linier pada
Bidang 2D Dengan Menggunakan OpenGL API”



Oleh :

Ricky Kennedy -13516105

Nicholas Wijaya - 13516121

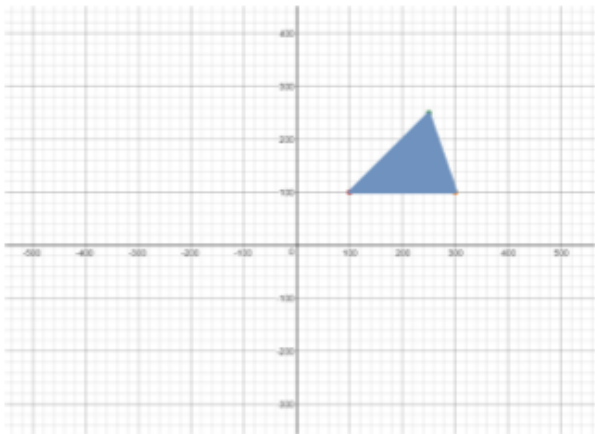
BAB I DESKRIPSI MASALAH

Pada tugas kali ini, mahasiswa diminta membuat program yang mensimulasikan transformasi linier untuk melakukan operasi translasi, refleksi, dilatasi, rotasi, dan sebagainya pada sebuah bidang 2D. Bidang dibuat dengan mendefinisikan sekumpulan titik sudut lalu membuat bidang dari titik-titik tersebut. Program akan memiliki dua buah window, window pertama (command prompt) berfungsi untuk menerima input dari user, sedangkan window kedua (GUI) berfungsi untuk menampilkan output berdasarkan input dari user. Kedua window ini muncul ketika user membuka file executable. Saat program baru mulai dijalankan, program akan menerima input N, yaitu jumlah titik yang akan diterima. Berikutnya, program akan menerima input N buah titik tersebut (pasangan nilai x dan y). Setelah itu program akan menampilkan output sebuah bidang yang dibangkitkan dari titik-titik tersebut. Selain itu juga ditampilkan dua buah garis, yaitu sumbu x dan sumbu y. Nilai x dan y memiliki rentang minimal 500 pixel dan maksimum 500 pixel. Pastikan window GUI yang Anda buat memiliki ukuran yang cukup untuk menampilkan kedua sumbu dari ujung ke ujung. Berikutnya, program dapat menerima input yang didefinisikan pada tabel dibawah.

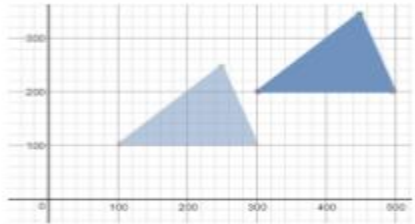
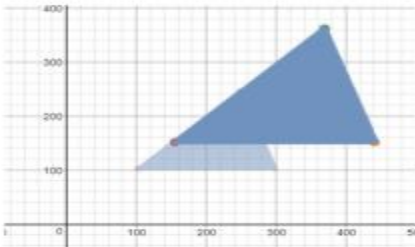
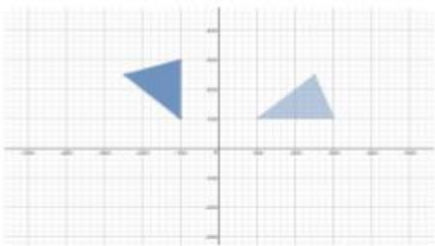
Input	Keterangan
<code>translate <dx> <dy></code>	Melakukan translasi objek dengan menggeser nilai x sebesar dx dan menggeser nilai y sebesar dy .
<code>dilate <k></code>	Melakukan dilatasi objek dengan faktor scaling k .
<code>rotate <deg> <a> </code>	Melakukan rotasi objek secara berlawanan arah jarum jam sebesar deg derajat terhadap titik a,b
<code>reflect <param></code>	Melakukan pencerminan objek. Nilai <i>param</i> adalah salah satu dari nilai-nilai berikut: $\mathbf{x}, \mathbf{y}, \mathbf{y}=\mathbf{x}, \mathbf{y}=-\mathbf{x}$, atau (\mathbf{a}, \mathbf{b}) . Nilai (a,b) adalah titik untuk melakukan pencerminan terhadap.
<code>shear <param> <k></code>	Melakukan operasi <i>shear</i> pada objek. Nilai <i>param</i> dapat berupa x (terhadap sumbu x) atau y (terhadap sumbu y). Nilai k adalah faktor <i>shear</i> .
<code>stretch <param> <k></code>	Melakukan operasi <i>stretch</i> pada objek. Nilai <i>param</i> dapat berupa x (terhadap sumbu x) atau y (terhadap sumbu y). Nilai k adalah faktor <i>stretch</i> .
<code>custom <a> <c> <d></code>	Melakukan transformasi linier pada objek dengan matriks transformasi sebagai berikut: $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$
<code>multiple <n></code> ... // input 1 ... // input 2 // input n	Melakukan transformasi linier pada objek sebanyak n kali berurutan. Setiap baris input 1.. n dapat berupa <i>translate</i> , <i>rotate</i> , <i>shear</i> , <i>dll</i> tetapi bukan <i>multiple</i> , <i>reset</i> , <i>exit</i> .
<code>reset</code>	Mengembalikan objek pada kondisi awal objek didefinisikan.
<code>exit</code>	Keluar dari program.

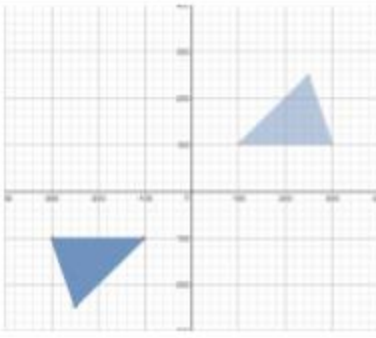
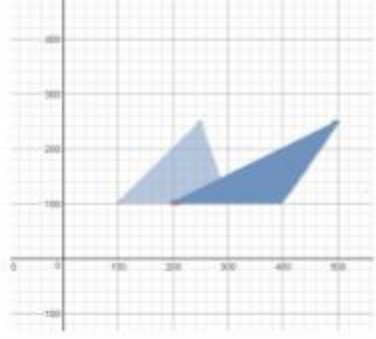
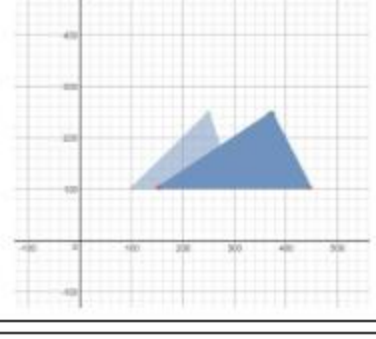
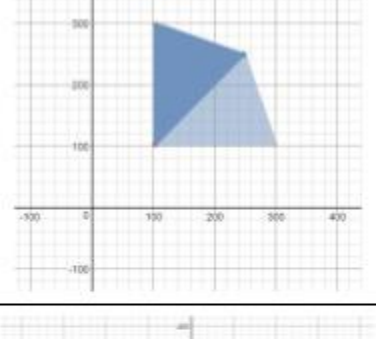
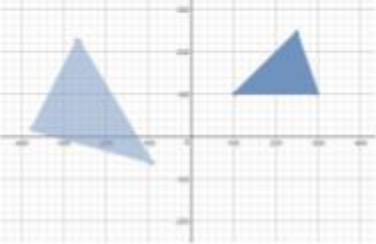
Contoh I/O Program :

Saat Program Pertama kali dijalankan dan meminta masukan jumlah input titik N dan koor (x,y) sebanyak N

Input	Output
<pre> 3 100,100 250,250 300,100 </pre>	

Catatan: Perhatikan bahwa gambar bidang yang transparan menunjukkan kondisi bidang sebelum input diberi, sedangkan bidang yang tidak transparan menunjukkan kondisi bidang setelah program mengeksekusi operasi dari input (bidang yang transparan tidak ditampilkan pada program).

Input	Output
<pre> translate 200 100 </pre>	
<pre> dilate 1.5 </pre>	
<pre> rotate 90 0 0 </pre>	

<p>reflect (0,0)</p>	
<p>shear x 1</p>	
<p>stretch x 1.5</p>	
<p>custom 0 1 1 0</p>	
<p>reset</p>	

BAB II DASAR TEORI

2.1. Transformasi Linear

Jika V dan W adalah ruang vektor dan F adalah sebuah fungsi yang mengasosiasikan sebuah vektor yang unik di dalam W dengan sebuah vektor di dalam V , maka kita mengatakan F memetakan V ke dalam W , dan kita menuliskan $F : V \rightarrow W$. Lebih lanjut lagi, jika F mengasosiasikan vektor \underline{w} dengan vektor \underline{v} , maka kita menuliskan $\underline{w} = F(\underline{v})$ dan kita mengatakan bahwa \underline{w} adalah bayangan dari \underline{v} di bawah F .

Untuk melukiskannya, maka jika $\underline{v} = (x, y)$ adalah sebuah vektor di dalam R^2 , maka rumus :

$$F(\underline{v}) = (x, x + y, x - y) \quad (4.1)$$

mendefinisikan sebuah fungsi yang memetakan R^2 ke dalam R^3 .

Khususnya, jika $\underline{v} = (1, 1)$, maka $x = 1$ dan $y = 1$, sehingga bayangan dari \underline{v} di bawah F adalah $F(\underline{v}) = (1, 2, 0)$.

Definisi.

Jika $F : V \rightarrow W$ adalah sebuah fungsi dari ruang vektor V ke dalam ruang vektor W , maka F dinamakan transformasi linear jika :

- (i) $F(\underline{u} + \underline{v}) = F(\underline{u}) + F(\underline{v})$ untuk semua vektor \underline{u} dan \underline{v} di dalam V .
- (ii) $F(k\underline{u}) = k F(\underline{u})$ untuk semua vektor \underline{u} di dalam V dan semua skalar k .

Untuk melukiskannya, misalkan $F : R^2 \rightarrow R^3$ adalah fungsi yang didefinisikan oleh (4.1).

Jika $\underline{u} = (x_1, y_1)$ dan $\underline{v} = (x_2, y_2)$, maka $\underline{u} + \underline{v} = (x_1 + x_2, y_1 + y_2)$, sehingga :

$$\begin{aligned} F(\underline{u} + \underline{v}) &= (x_1 + x_2, [x_1 + x_2] + [y_1 + y_2], [x_1 + x_2] - [y_1 + y_2]) \\ &= (x_1, x_1 + y_1, x_1 - y_1) + (x_2, x_2 + y_2, x_2 - y_2) \end{aligned}$$

$$F(\underline{u} + \underline{v}) = F(\underline{u}) + F(\underline{v})$$

Juga, jika k adalah sebuah skalar, $k\underline{u} = (kx_1, ky_1)$, sehingga

$$\begin{aligned} F(k\underline{u}) &= (kx_1, kx_1 + ky_1, kx_1 - ky_1) \\ &= k(x_1, x_1 + y_1, x_1 - y_1) \\ &= k F(\underline{u}) \end{aligned}$$

Jadi F adalah sebuah transformasi linear.

Jika $F : V \rightarrow W$ adalah sebuah transformasi linear, maka untuk sebarang v_1 dan v_2 di dalam V dan

sebarang k_1 dan k_2 , kita memperoleh :

$$F(k_1 \underline{v}_1 + k_2 \underline{v}_2) = F(k_1 \underline{v}_1) + F(k_2 \underline{v}_2) = k_1 F(\underline{v}_1) + k_2 F(\underline{v}_2)$$

Demikian juga, jika $\underline{v}_1, \underline{v}_2, \dots, \underline{v}_n$ adalah vektor-vektor di dalam V dan k_1, k_2, \dots, k_n adalah skalar, maka :

$$F(k_1 \underline{v}_1 + k_2 \underline{v}_2 + \dots + k_n \underline{v}_n) = k_1 F(\underline{v}_1) + k_2 F(\underline{v}_2) + \dots + k_n F(\underline{v}_n)$$

2.2. Matriks Transformasi

kolom dari matriks transisi dari basis lama ke basis baru adalah vektor koordinat dari basis lama relatif terhadap basis baru.

2.3 Open GL

Open Graphics Library (OpenGL) adalah API yang berfungsi untuk melakukan rendering grafik 2D dan 3D. OpenGL bersifat cross-language, cross-platform, dan open source. OpenGL umumnya digunakan untuk melakukan interaksi dengan GPU (graphics processing unit) untuk mencapai hasil render yang diakselerasi dengan hardware. Anda diharapkan untuk melakukan eksplorasi penggunaan OpenGL

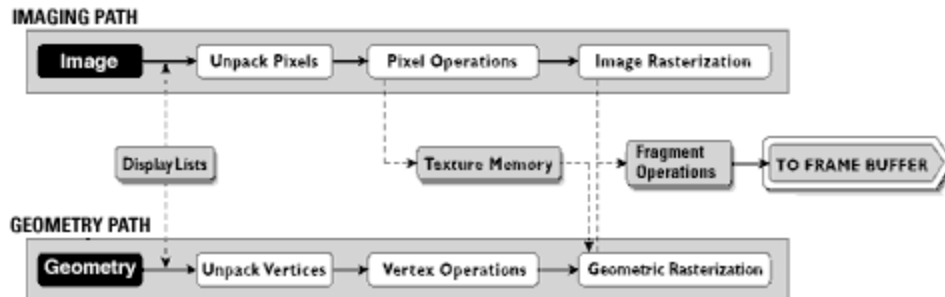
A. Standar Grafis yang Paling Diadopsi Secara Adil

OpenGL adalah lingkungan utama untuk mengembangkan aplikasi grafis 2D dan 3D portabel dan interaktif. Sejak diperkenalkan pada tahun 1992, OpenGL telah menjadi antarmuka pemrograman aplikasi grafis 2D dan 3D yang paling banyak digunakan dan didukung oleh industri, membawa ribuan aplikasi ke berbagai platform komputer. OpenGL memupuk inovasi dan mempercepat pengembangan aplikasi dengan menggabungkan seperangkat rendering, pemetaan tekstur, efek khusus, dan fungsi visualisasi lainnya yang hebat. Pengembang dapat memanfaatkan kekuatan OpenGL di semua platform desktop dan workstation populer, memastikan penerapan aplikasi yang luas.

B. Kualitas dan Kinerja Visual Tinggi

Setiap aplikasi komputasi visual yang membutuhkan kinerja maksimal - mulai dari animasi 3D sampai CAD hingga simulasi visual - dapat memanfaatkan kemampuan OpenGL berkualitas tinggi dan berkinerja tinggi. Kemampuan ini memungkinkan pengembang di pasar yang beragam seperti penyiaran, CAD / CAM / CAE, hiburan, pencitraan medis, dan virtual reality untuk menghasilkan dan menampilkan grafik 2D dan 3D yang sangat menarik.

C. The OpenGL Visualization Programming Pipeline



OpenGL operates on image data as well as geometric primitives.

D. Available Everywhere

Didukung di semua workstation UNIX®, dan dikirim standar dengan setiap PC Windows 95/98/2000 / NT dan MacOS, tidak ada API grafis lainnya yang beroperasi pada platform perangkat keras dan lingkungan perangkat lunak yang lebih luas. OpenGL berjalan pada setiap sistem operasi utama termasuk Mac OS, OS / 2, UNIX, Windows 95/98, Windows 2000, Windows NT, Linux, OPENStep, dan BeOS; Ini juga bekerja dengan setiap sistem windowing utama, termasuk Win32, MacOS, Presentation Manager, dan X-Window System. OpenGL dapat dipanggil dari Ada, C, C ++, Fortran, Python, Perl dan Java dan menawarkan kebebasan penuh dari protokol jaringan dan topologi.

E. Architected for Flexibility and Differentiation: Extensions

Meskipun spesifikasi OpenGL mendefinisikan sebuah pipeline pemrosesan grafis tertentu, vendor platform memiliki kebebasan untuk menyesuaikan implementasi OpenGL tertentu untuk memenuhi biaya sistem dan kinerja yang unik. Panggilan individu dapat dilakukan pada perangkat keras khusus, dijalankan sebagai rutinitas perangkat lunak pada CPU sistem standar, atau diterapkan sebagai kombinasi dari rutinitas perangkat keras dan perangkat lunak yang didedikasikan. Fleksibilitas implementasi ini berarti akselerasi perangkat keras OpenGL dapat berkisar dari rendering sederhana hingga geometri penuh dan tersedia secara luas dalam segala hal mulai dari komputer murah hingga workstation kelas atas dan superkomputer. Pengembang aplikasi yakin hasil tampilan konsisten terlepas dari implementasi platform lingkungan OpenGL.

Dengan menggunakan mekanisme ekstensi OpenGL, pengembang perangkat keras dapat membedakan produk mereka dengan mengembangkan ekstensi yang memungkinkan pengembang perangkat lunak mengakses kinerja dan inovasi teknologi tambahan.

Banyak ekstensi OpenGL, serta ekstensi untuk API terkait seperti GLU, GLX, dan WGL, telah ditentukan oleh vendor dan grup vendor. OpenGL Extension Registry dikelola oleh SGI dan berisi spesifikasi untuk semua ekstensi yang dikenal, yang ditulis sebagai modifikasi pada dokumen spesifikasi yang sesuai. Registri juga mendefinisikan konvensi penamaan, pedoman untuk membuat ekstensi baru dan menulis spesifikasi ekstensi yang sesuai, dan dokumentasi terkait lainnya.

BAB III IMPLEMENTASI PROGRAM

PADA BAHASA PEMOGRAMAN PYTHON

Implementasi OpenGL diperlukan pra persyaratan yaitu kita harus meng-*import* beberapa library yang terdapat pada python agar OpenGL dapat berjalan dengan baik dan berikut adalah library yang kita gunakan :

```
from OpenGL.GL import*
from OpenGL.GLUT import*
from OpenGL.GLU import*
from tr2d import *
import numpy as np
import serial
import os
import sys
import time
```

Karena pada tugas kali ini memerlukan OpenGL maka library yang pertama di-*import* adalah OpenGL.GL, OpenGL.GLUT, dan OpenGL.GLU ketiga library ini untuk menjalankan OpenGL pada python. Kedua melakukan import terhadap semua fungsi yang tr2d, tr2d adalah file yang berisi fungsi-fungsi yang digunakan untuk mengoperasikan titik-titik yang dimasukan. Dan *import-import* berikutnya untuk membantu mempermudah memproses data yang masuk seperti numpy dipergunakan untuk memasukan titik kedalam array, time untuk melakukan animasi dan lain sebagainya.

Untuk memudahkan memogram tugas OpenGL ini, kami memogramnya secara modular. dan membagi program kedalam beberapa fungsi sebagai berikut :

```
def InitGL(width, height): ***
def GLSWindow(): ***
def drawsumbu(): ***
def input_titik(): ***
def DrawShape(): ***
def DrawGLScene(): ***
def input_function(): ***
def main(): ***
```

- Fungsi yang pertama InitGL adalah fungsi yang digunakan untuk melakukan inisialisasi GUI yang akan dibuat.
- Fungsi kedua GLSWindow adalah fungsi untuk membuat GUI menjalankan program pada GUI sesuai dengan kode program.

- Fungsi ketiga drawsumbu adalah fungsi yang dibuat untuk membuat diagram kartesius pada saat ditampilkan pada GUI.
- Fungsi keempat input_titik adalah fungsi yang dijalankan saat pertama kali dan diperuntukan menerima banyak titik dan letak titik yang berupa masukan dari user.
- Fungsi kelima DrawShape adalah fungsi yang diperuntukan menggambar bentuk atau area dari titik-titik setelah menerima masukan dari user atau juga menggambar area setelah dieksekusi dengan fungsi-fungsi yang telah disediakan.
- Fungsi keenam DrawGLScene adalah fungsi yang diperuntukan menampilkan hasil program pada GUI dan juga melakukan proses animasi.
- Fungsi ketujuh input_function adalah fungsi yang diperuntukan untuk menerima masukan dari user yang berupa fungsi untuk mengolah titik-titik awal yang telah dimasukan oleh user diawal program dan implementasi fungsi-fungsi tersebut terdapat pada file “tr2d.py”.

File “tr2d.py” adalah file yang berisi fungsi-fungsi yang memproses titik awal sesuai dengan masukan pada fungsi “input_function” dan berikut adalah daftar fungsi yang disediakan untuk memgolah titik-titik sesuai dengan keinginan user.

```
def translate(vin,dx,dy):
#dx dan dy merupakan float
#vin adalah list yang menampung titik (Matriks), dx perubahan Koordinat X dan dy perubahan koordinat Y
#mengembalikan matriks baru yang berisi vin yang sudah translasi ***

def dilate(vin,k):
#vin dilatasi dengan faktor skala k
#mengembalikan matriks baru yang berisi vin yang sudah dilatasi ***

def rotate(vin,deg,a,b):
#mengembalikan matriks baru yang berisi vin yang sudah dirotasi pada titik (a,b) dan deg derajat ***

def reflect(vin,param):
#mengembalikan matriks baru yang merupakan vin yang sudah direfleksikan dengan x,y,y=x,y=-x,atau titik (a,b) ***

def shear(vin,par,k):
#mengembalikan matriks baru yang merupakan vin yang telah digusur terhadap sumbu <par> dengan faktor gusuran <k> ***

def stretch(vin,par,k):
#mengembalikan matriks baru yang merupakan vin yang telah diregangkan terhadap sumbu <par> dengan faktor regangan <k> ***

def custom(vin,a,b,c,d):
#mengembalikan matriks baru yang merupakan vin yang telah dikalikan dengan matriks transformasi [[a,b],[c,d]] ***

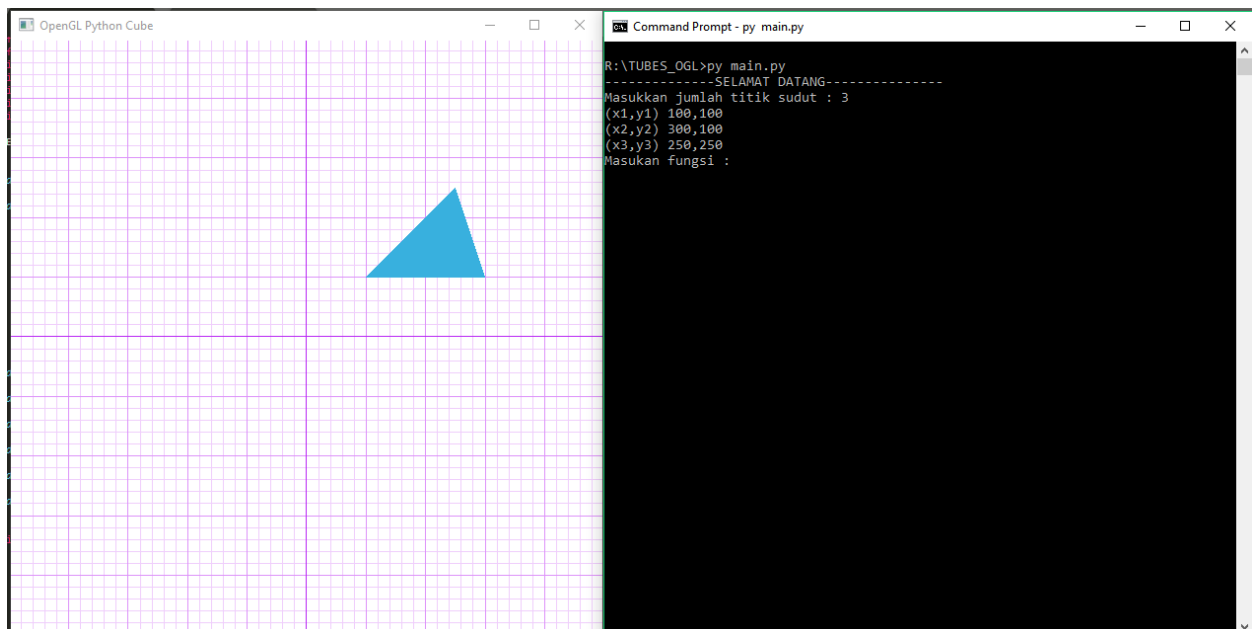
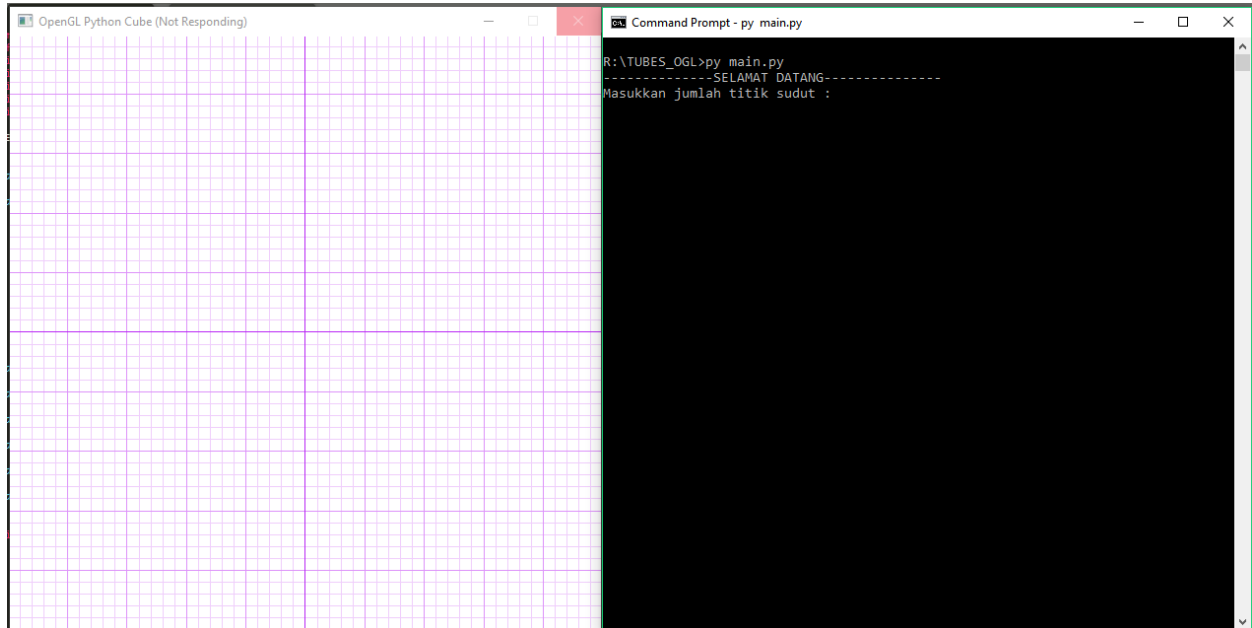
def multiple(vin,n):
#Melakukan transformasi linier pada vin sebanyak n kali berurutan.
#Setiap baris input 1..n dapat berupa translate, rotate, shear, dll tetapi bukan multiple, reset, exit *** s
```

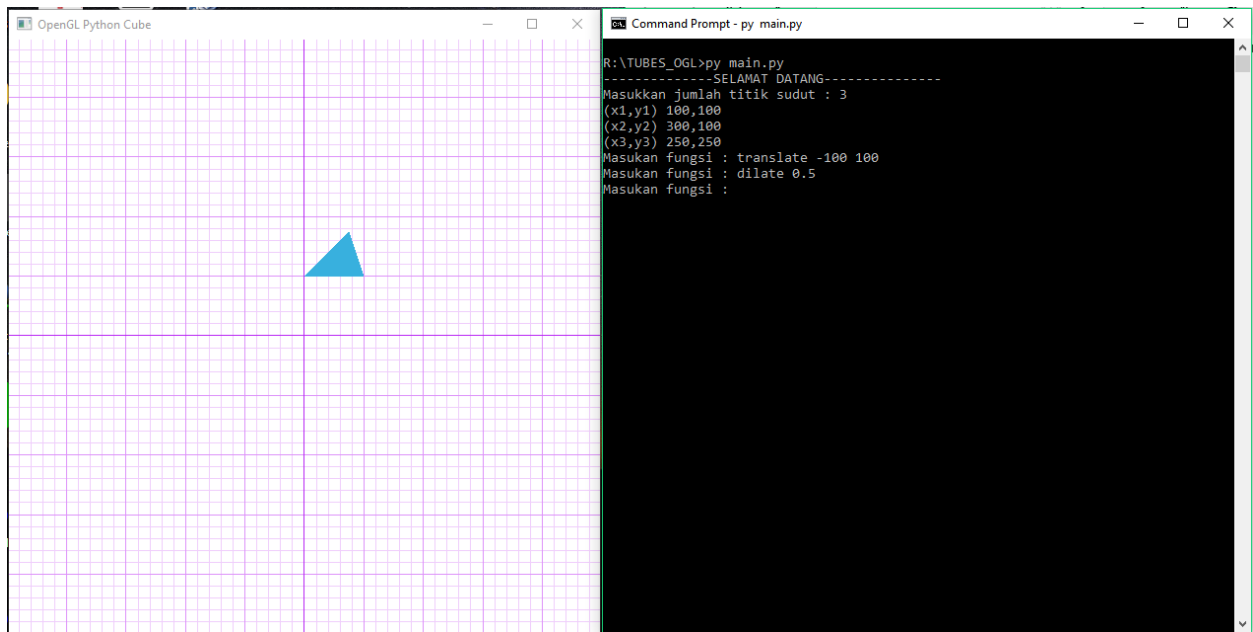
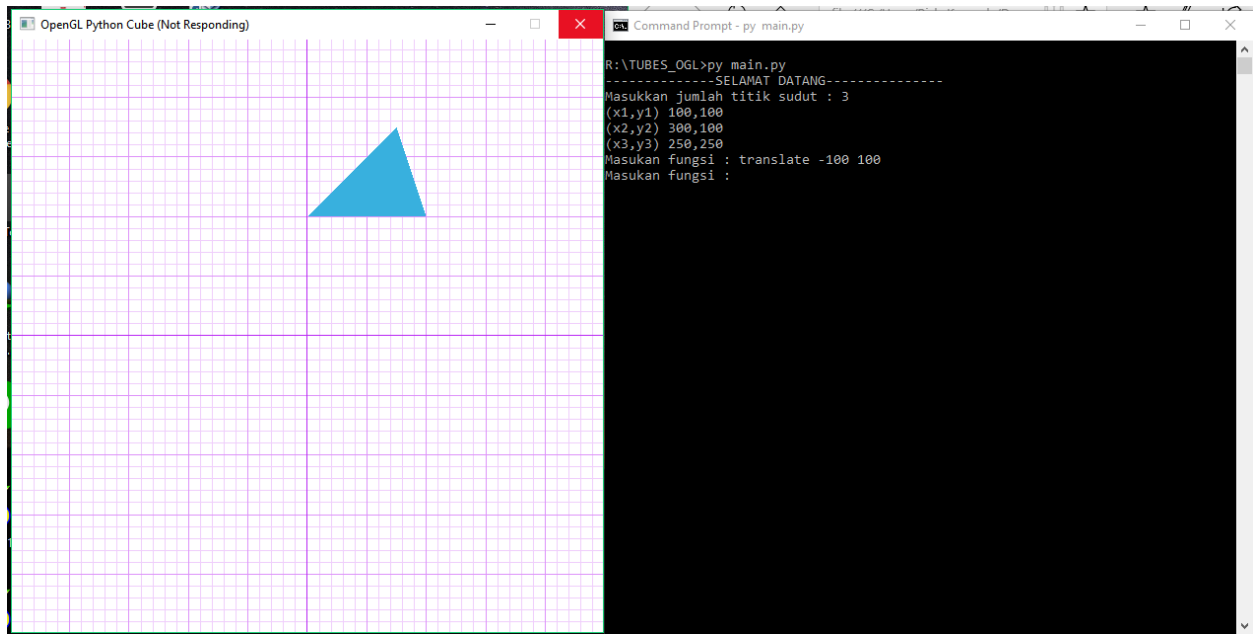
- Fungsi translate adalah fungsi yang menerima masukan vin, dx, dan dy dimana dx dan dy merupakan variable bertipe float (desimal). Vin adalah list yang menampung titik (Matriks), dx perubahan koordinat X dan dy perubahan koordinat Y mengembalikan matriks baru yang berisi vin yang sudah translasi.

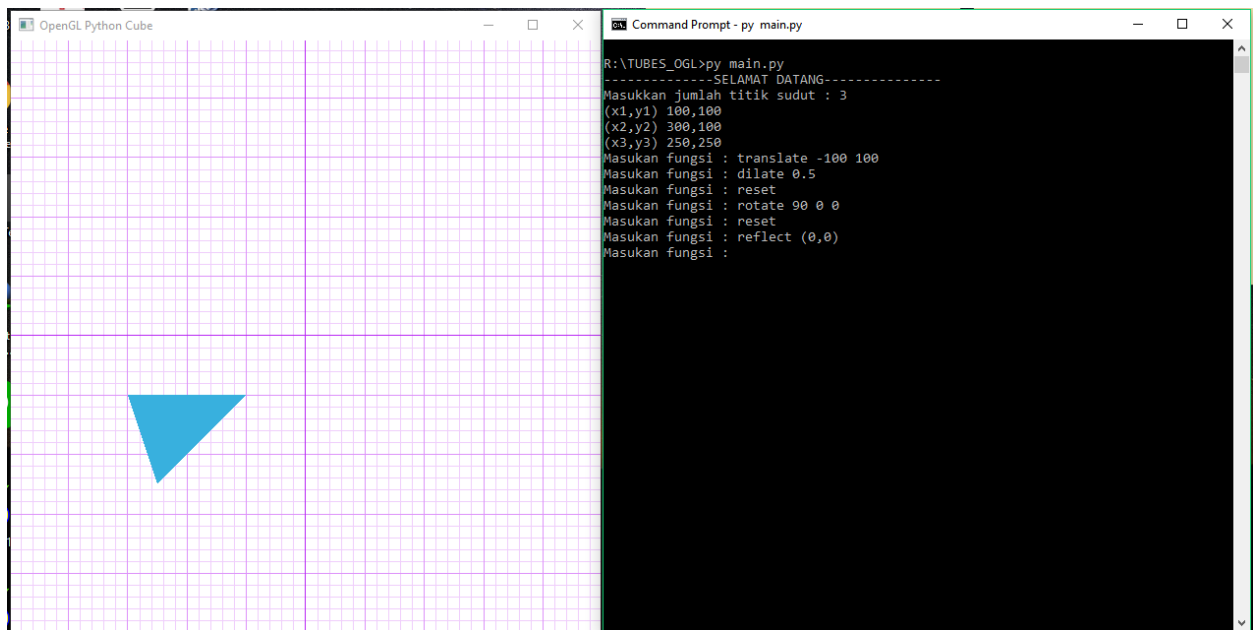
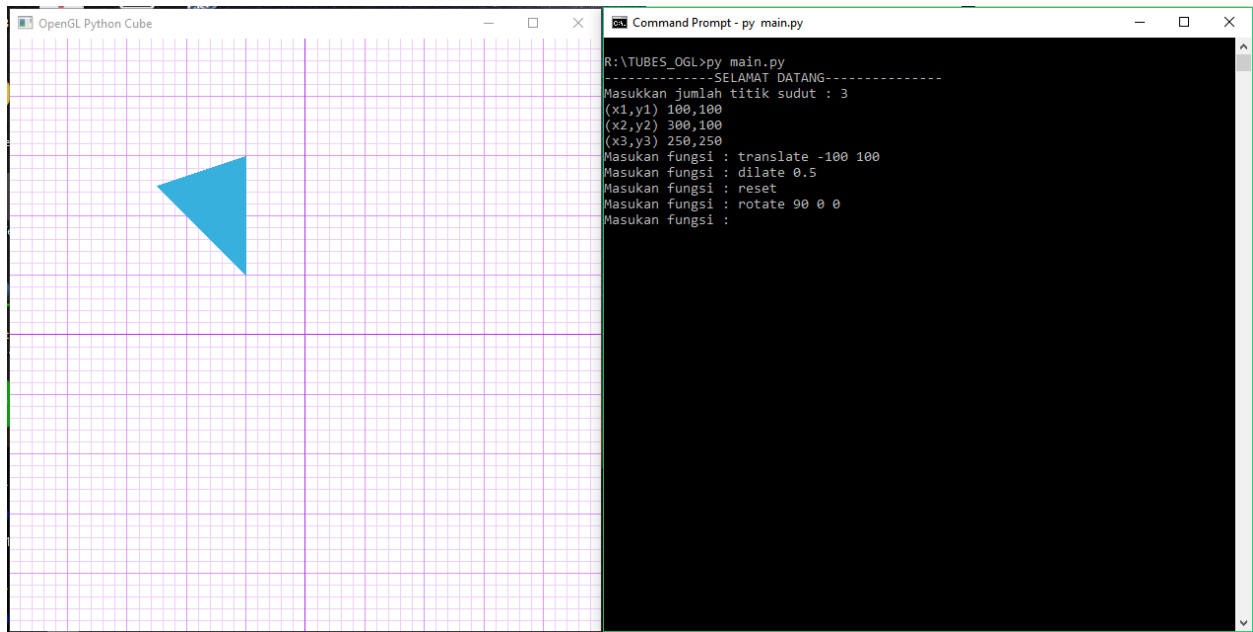
- Fungsi dilate adalah fungsi untuk memilatasikan matriks dengan faktor k.
- Fungsi rotate adalah fungsi yang mengembalikan matriks baru yang merupakan vin yang telah dirotasikan dengan derajat sebesar $\langle \text{deg} \rangle$ dan pada titik (a,b).
- Fungsi reflect adalah fungsi yang mengembalikan matriks baru yang merupakan vin yang sudah direfleksi dengan x, y, $y=x$, $y=-x$, atau titik (a,b)
- Fungsi shear adalah fungsi yang mengembalikan matriks baru yang merupakan vin yang telah digusur terhadap sumbu x atau y.
- Fungsi stretch adalah fungsi yang mengembalikan matriks baru yang merupakan vin yang telah diregangkan terhadap terhadap sumbu x ataupun y.
- Fungsi custom adalah fungsi yang mengembalikan matriks baru yang merupakan vin yang telah dikalikan dengan matriks transformasi inputan pengguna.
- Fungsi multiple adalah fungsi yang mengembalikan matriks baru yang merupakan vin yang telah ditransformasi sebanyak $\langle \text{input pengguna} \rangle$ kali secara berurutan.

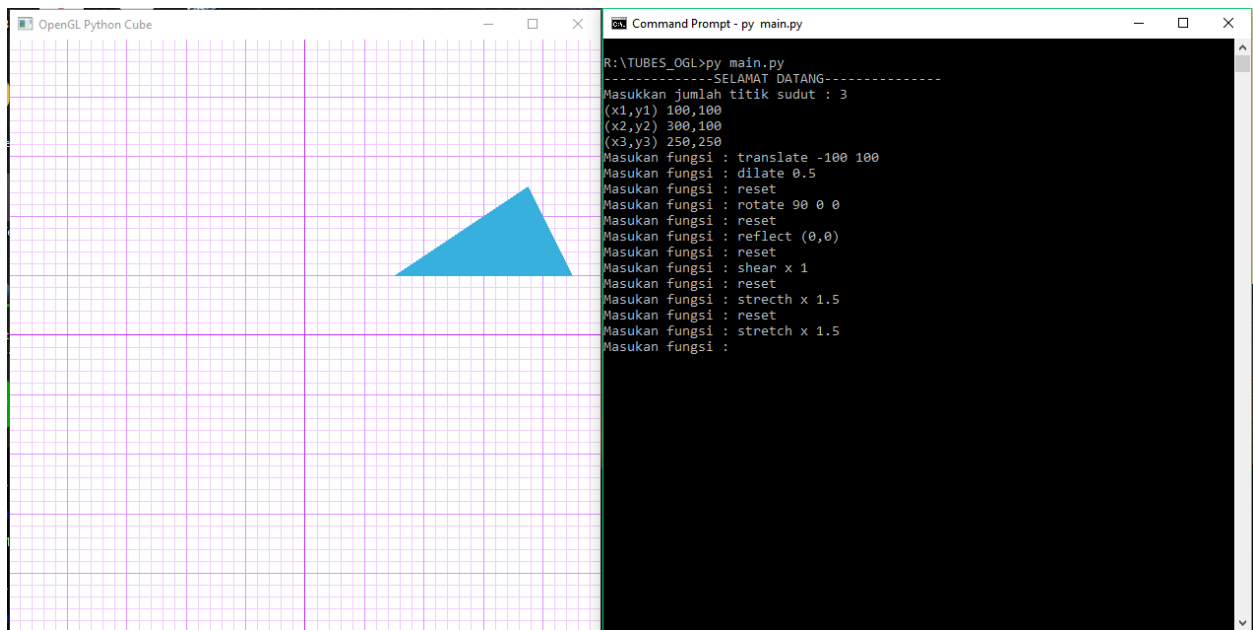
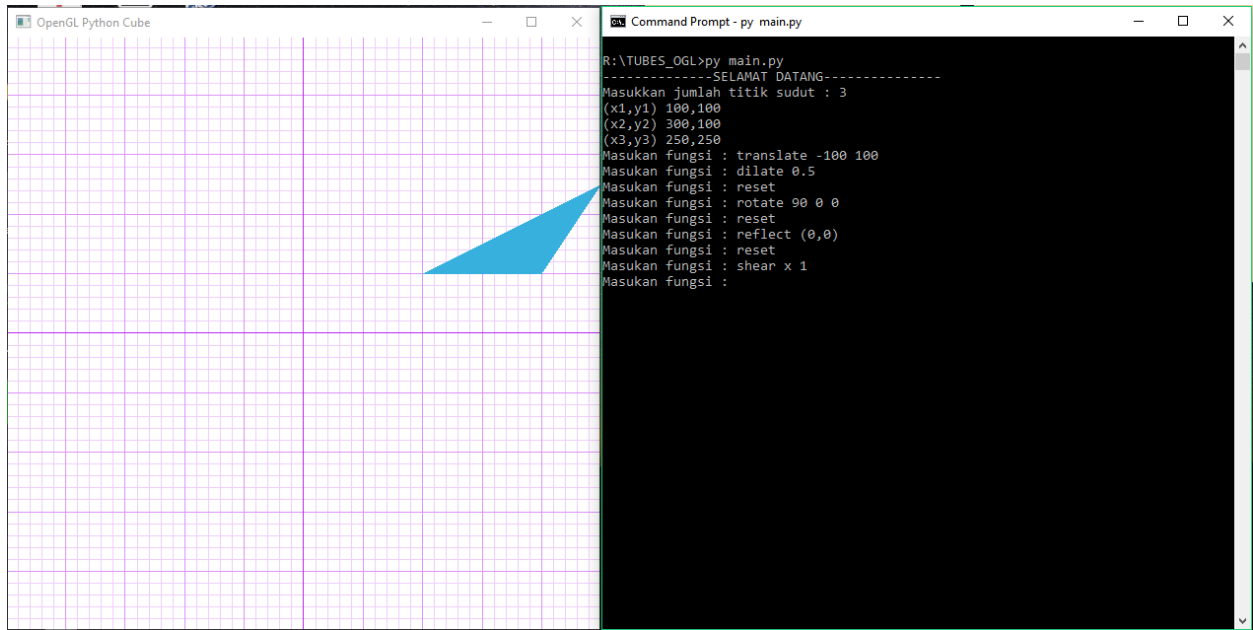
BAB IV EKSPERIMEN PADA PROGRAM

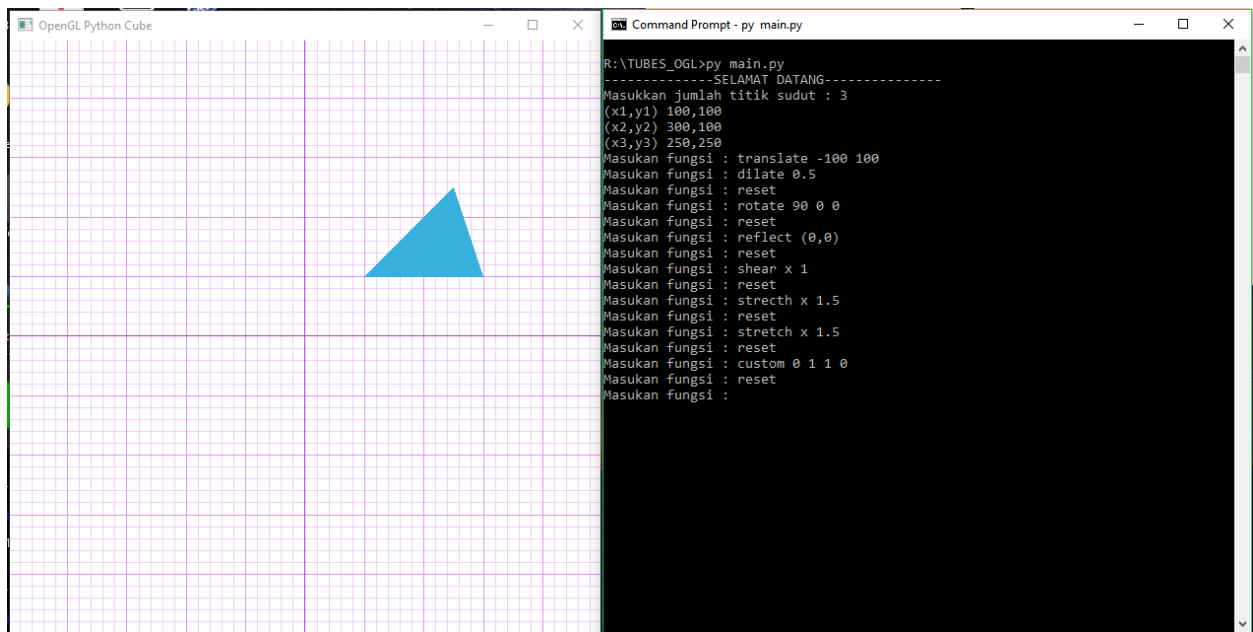
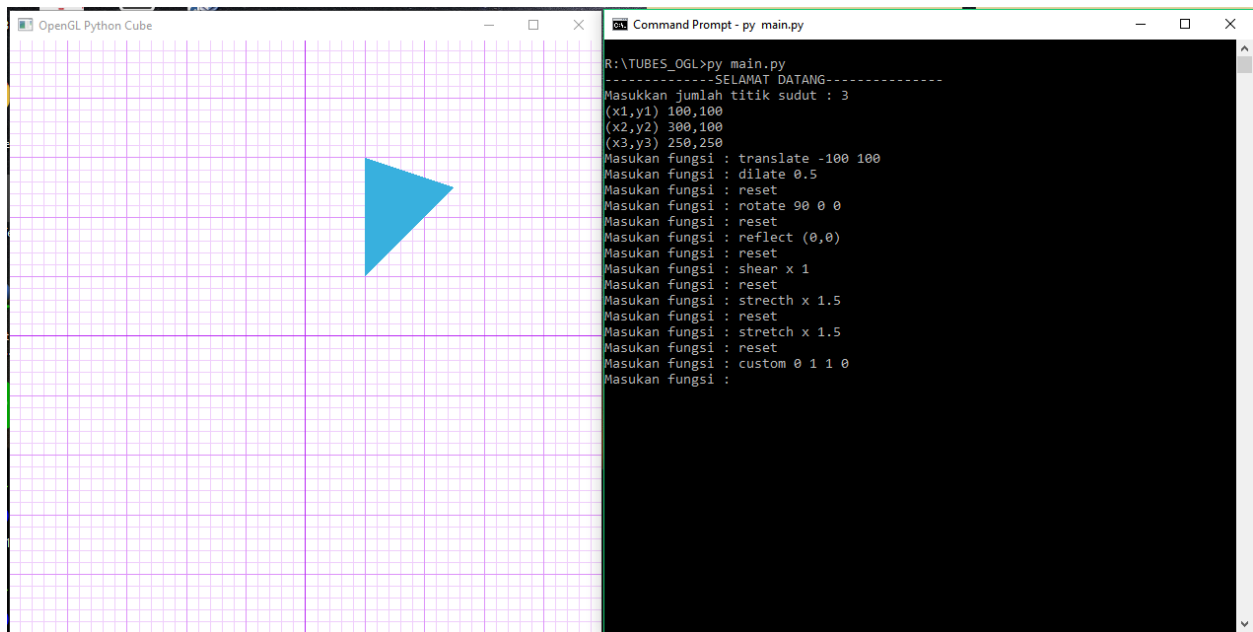
Semua eksekusi titiku dilakukan dari titik awal dan tidak diproses secara berlanjut.

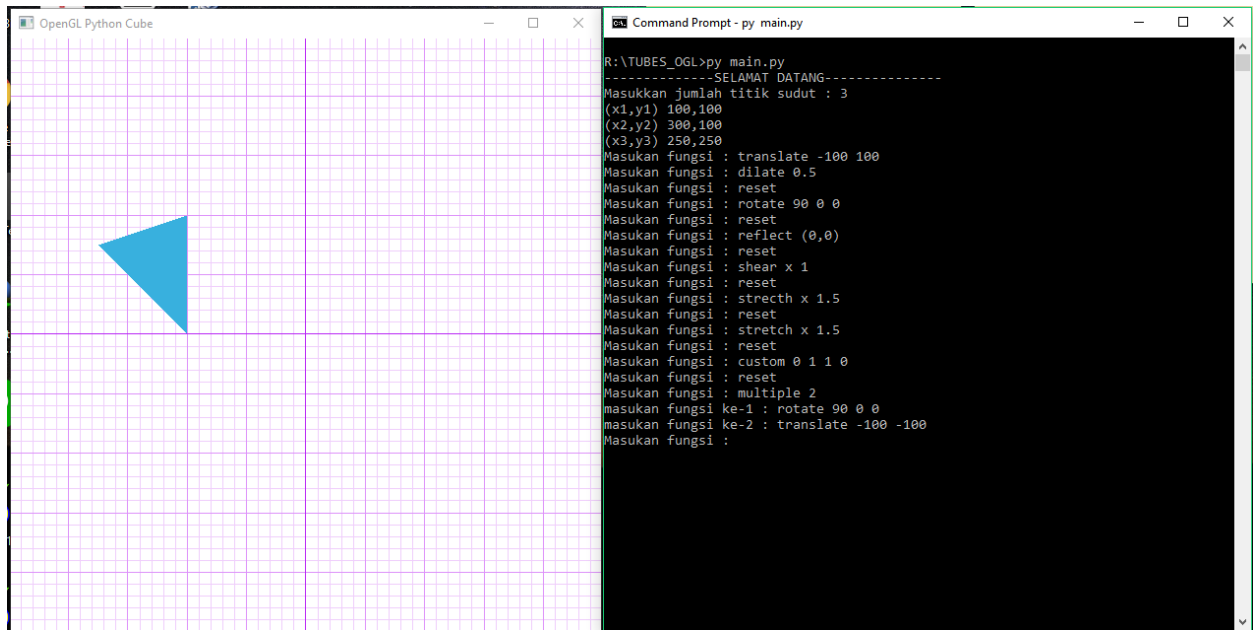












```
Command Prompt
R:\TUBES_OGL>py main.py
-----SELAMAT DATANG-----
Masukkan jumlah titik sudut : 3
(x1,y1) 100,100
(x2,y2) 300,100
(x3,y3) 250,250
Masukkan fungsi : translate -100 100
Masukkan fungsi : dilate 0.5
Masukkan fungsi : reset
Masukkan fungsi : rotate 90 0 0
Masukkan fungsi : reset
Masukkan fungsi : reflect (0,0)
Masukkan fungsi : reset
Masukkan fungsi : shear x 1
Masukkan fungsi : reset
Masukkan fungsi : stretch x 1.5
Masukkan fungsi : reset
Masukkan fungsi : stretch x 1.5
Masukkan fungsi : reset
Masukkan fungsi : custom 0 1 1 0
Masukkan fungsi : reset
Masukkan fungsi : multiple 2
masukan fungsi ke-1 : rotate 90 0 0
masukan fungsi ke-2 : translate -100 -100
Masukkan fungsi : reset
Masukkan fungsi : exit
SAMPAI JUMPA ;)

R:\TUBES_OGL>
```

BAB V Kesimpulan dan Saran

5.1 Kesimpulan

Dalam proses pembuatan program untuk mensimulasikan transformasi linier pada bidang 2D beserta animasinya, kami menemukan bahwa persoalan transformasi geometri dapat dengan mudah diselesaikan dengan metode perkalian menggunakan matriks transformasi. Kami juga memanfaatkan Antarmuka Pemrograman Aplikasi (API) OpenGL dan bahasa pemrograman Python, untuk tampilan proses transformasi linier terhadap suatu bidang. Program yang kami buat lancar dan telah memenuhi spesifikasi yang telah diberikan. Program kami juga dapat menampilkan animasi proses transformasi yang dilakukan.

5.2 Saran

Saran yang dapat kami sampaikan adalah berikan tutorial lebih lanjut mengenai cara menggunakan OpenGL dalam kedua bahasa pemrograman yang diperbolehkan. Sekian saran dan kesimpulan yang dapat kami sampaikan, semoga laporan ini dapat berguna untuk ke depannya.