LAPORAN TUGAS BESAR IF2123:ALJABAR GEOMETRI

Simulasi Transformasi Linier pada Bidang 2D Dengan Menggunakan OpenGL API

Kelas 02

oleh

Manasye Shousen Bukit	13516122
Regi Arjuna Purba	13516149

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

2017

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 maksikum 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**
translate <dx> <dy>	Melakukan translasi objek dengan menggeser nilai x sebesar *dx* dan menggeser nilai y sebesar *dy*.
dilate <k>	Melakukan dilatasi objek dengan faktor scaling *k*.
rotate <deg> <a> <b>	Melakukan rotasi objek secara berlawanan arah jarum jam sebesar *deg* derajat terhadap titik *a*,*b*
reflect <param>	Melakukan pencerminan objek. Nilai *param* adalah salah satu dari nilai-nilai berikut: **x**, **y**, **y=x**, **y=-x**, atau **(a,b)**. Nilai (a,b) adalah titik untuk melakukan pencerminan terhadap.

shear <param> <k>	Melakukan operasi *shear* pada objek. Nilai *param* dapat berupa *x* (terhadap sumbu x) atau *y* (terhadap sumbu y). Nilai *k* adalah faktor *shear*.
stretch <param> <k>	Melakukan operasi *stretch* pada objek. Nilai *param* dapat berupa *x* (terhadap sumbu x) atau *y* (terhadap sumbu y). Nilai *k* adalah faktor *stretch*.
custom <a> <b> <c> <d>	Melakukan transformasi linier pada objek dengan matriks transformasi
multiple <n> ... // input 1 ... // input 2 ... ... // input n	Melakukan transformasi linier pada objek sebanyak *n* kali berurutan. Setiap baris input 1..*n* dapat berupa *translate, rotate, shear, dll* tetapi bukan *multiple, reset, exit*.
reset	Mengembalikan objek pada kondisi awal objek didefinisikan.
exit	Keluar dari program.

BAB II
TEORI SINGKAT

2.1 Transformasi Linier

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 dapat memetakan V ke dalam W, dan kita menuliskan Lebih lanjut lagi, jika F mengasosiasikan vektor w dengan vektor v, maka kita menuliskan w= F(v) dan kita mengatakan bahwa w adalah bayangan dari v di bawah F.

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

$$F(v) = (x , x + y , x - y)$$

mendefinisikan sebuah fungsi yang memetakan R^2 ke dalam R^3. Khususnya, jika v = (1,1) , maka x = 1 dan y = 1 , sehingga bayangan dari v di bawah F adalah F(v) = (1, 2, 0).

Definisi

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

(i) F(u+ v) = F(u) + F(v) untuk semua vektor u dan v di dalam V.
(ii) F(ku) = k F(u) untuk semua vektor u di dalam V dan semua skalar k.

Untuk melukiskannya, misalkan F : $R^2 \rightarrow R^3$ adalah fungsi yang didefinisikan oleh
Jika u = (x1, y1) dan v= (x2, y2), maka u+ v= (x1+ x2, y1 + y2), sehingga :

$$F(u+ v) = (x1+x2, [x1+ x2] + [y1+ y2], [x1+ x2] -[y1+ y2])$$
$$= (x1, x1+ y1, x1-y1) + (x2, x2+ y2,x2-y2)$$
$$F(u+v) = F(u) + F(v)$$

Juga , jika k adalah sebuah skalar , k u= (kx1, ky1), sehingga

$$F(k\,u) = (kx1, kx1+ky1, kx1-ky1)$$
$$= k\,(x1, x1+y1,x1-y1)$$
$$= k\,F(u)$$

Jadi F adalah sebuah transformasi linear.

Jika F : V →W adalah sebuah transformasi linear, maka untuk sebarang v1 dan v2 di dalam V dan sebarang k1 dan k2, kita memperoleh :

$$F(k1v1+ k2v2) = F(k1v1) + F(k2v2) = k1F(v1) + k2F(v2)$$

Demikian juga, jika v1, v2, ... , vn adalah vektor - vektor di dalam V dan k1, k2, ... , kn adalah skalar, maka

$$F(k1v1+ k2v2+ \dots + knvn) = k1F(v1) + k2F(v2) + \dots + knF(vn)$$

2.2 Matriks Transformasi

Input	**Matriks Transformasi**
translate	$\begin{pmatrix} a \\ b \end{pmatrix}$
dilate	$\begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}$
rotate	$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$
reflect	$R_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ $R_y = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$
shear	X'= $\begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}$ Y'= $\begin{pmatrix} 1 & 0 \\ k & 1 \end{pmatrix}$
stretch	X'= $\begin{pmatrix} k & 0 \\ 0 & 1 \end{pmatrix}$ Y'= $\begin{pmatrix} 1 & 0 \\ 0 & k \end{pmatrix}$

BAB III

IMPLEMENTASI PROGRAM

```
# Inputing the number of points
nbelmt = input("Input number of point : ")

# Initialization of the list needed
vertices = []

# Set the vertices based on users inputs
for counter in range(nbelmt):
    vertices.append([])
    pointsx = input("x[" + str(counter) + "] is : ")
    pointsy = input("y[" + str(counter) + "] is : ")
    vertices[counter].append(float(pointsx)+float(originX))
    vertices[counter].append(float(pointsy)+float(originY))

vertices2 = copy.deepcopy(vertices) # use module copy to copy all variable inside list
```

Menerima inputan dari user jumlah titik dan koordinatnya.

```
# Function to draw the object
def draw():

    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)  # clear the screen
    glLoadIdentity()  # reset position

    refresh2d(width, height)
    drawAxis()

    glBegin(GL_POLYGON)
    glColor3f(0,0,1)
    for counter in range(len(vertices)):
        pointx = vertices[counter][0]
        pointy = vertices[counter][1]
        glVertex2f(pointx,pointy)
    glEnd()

    glutSwapBuffers()  # for double buffering

# Function to draw cartesian axis with grid in it
```

Fungsi yang akan digambar di jendela.Fungsi drawAxis menggambar koordinat dan GL_POLYGON untuk menggambar polygon sesuai masukan pengguna.

```
# Function to display the window with images in it
def window() :
    glutInit() # initializing glut
    glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE | GLUT_ALPHA | GLUT_DEPTH)
    glutInitWindowSize(width, height) # set window's size
    glutInitWindowPosition(0, 0) # set window's inital position
    window = glutCreateWindow("Tubes Algeo 2") # create window with title
    glClearColor(1.0, 1.0, 1.0, 1.0) # create a grey/gray background
    glutDisplayFunc(draw) # set draw function callback
    glutIdleFunc(draw)  # draw images all the time
    glutMainLoop()  # the loop

# Make a thread so we can input command while having the main drawing loop
start_new_thread(window, ())
```

Fungsi window untuk menampilkan layar GLUT dan start_new_thread untuk menampilkan layar GLUT berbarengan dengan command prompt.

```
# Function to multiply value of two matrixes
def multiply(matrix1, matrix2):
    result = []
    for i in range(len(matrix1)):
        result.append(0)
        for k in range(len(matrix2)):
            result[i] += matrix1[i][k] * matrix2[k]
    return result

# Function to add value of two matrixes
def add(matrix1, matrix2):
    result = []
    for i in range(len(matrix1)):
        result.append(matrix1[i] + matrix2[i])
    return result
```

2 fungsi untuk mengalikan dan menambahkan 2 buah matriks yang akan dipakai saat transformasi.

```
# Inputing the command and split it so we can later use each string as an instruction
command = raw_input().split()

while(True):
    # Action for translating
    if command[0] == 'translate':
        translate(float(command[1]),float(command[2]))

    # Action for dilating
    elif command[0] == 'dilate':
        dilate(float(command[1]))

    # Action for rotating
    elif command[0] == 'rotate':
        rotate(float(command[1]),float(command[2]),float(command[3]))

    # Action for reflecting
    elif command[0] == 'reflect':
        parameter = command[1]
        if ((parameter == 'x') or (parameter == 'y') or (parameter == 'y=x') or (parameter =='y=-x')):
            reflect(parameter)
        else :
            reflect2(parameter)

    # Action for shearing
    elif command[0] == 'shear':
        shear(command[1], float(command[2]))

    # Action for stretching
    elif command[0] == 'stretch':
        stretch(command[1], float(command[2]))

    # Action for customing
    elif command[0] == 'custom':
        custom(float(command[1]),float(command[2]),float(command[3]),float(command[4]))

    # Action for multiple action
```

```
# Action for multiple action
elif command[0] == 'multiple':
    N = int(command[1])
    cmd_inside = []

    # Looping through each line and get the command
    for i in range(N):
        cmd = raw_input().split()
        cmd_inside.append(cmd)

    for i in range(N):
        # Action for translating
        if cmd_inside[i][0] == 'translate':
            translate(float(cmd_inside[i][1]), float(cmd_inside[i][2]))

        # Action for dilating
        elif cmd_inside[i][0] == 'dilate':
            dilate(float(cmd_inside[i][1]))

        # Action for rotating
        elif cmd_inside[i][0] == 'rotate':
            rotate(float(cmd_inside[i][1]), float(cmd_inside[i][2]), float(cmd_inside[i][3]))

        # Action for reflecting
        elif cmd_inside[i][0] == 'reflect':
            parameter = cmd_inside[i][1]
            if ((parameter == 'x') or (parameter == 'y') or (parameter == 'y=x') or (parameter =='y=-x')):
                reflect(parameter)
            else :
                reflect2(parameter)

        # Action for shearing
        elif cmd_inside[i][0] == 'shear':
            shear(cmd_inside[i][1], float(cmd_inside[i][2]))

        # Action for stretching
        elif cmd_inside[i][0] == 'stretch':
            stretch(cmd_inside[i][1], float(cmd_inside[i][2]))

        # Action for customing
        elif cmd_inside[i][0] == 'custom':
            custom(float(cmd_inside[i][1]),float(cmd_inside[i][2]),float(cmd_inside[i][3]),float(cmd_inside[i][4]))
```

```
        # Action for wrong command
        else :
            print "You're inputting wrong command!"

# Action for reseting
elif command[0] == 'reset':
    reset()

# Action for exitting
elif command[0] == 'exit':
    # Print the title of the program
    print("-----------------------------------------------")
    print("|        Thanks for using this program       |")
    print("|       ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~      |")
    print("|              See you next time             |")
    print("-----------------------------------------------")
    os._exit(1) # exits the program without calling cleanup handlers, flushing stdio buffers, e

# Action for wrong command
else :
    print "You're inputting wrong command!"

# Inputing the command and split it so we can later use each string as an instruction
command = raw_input().split()
```

Looping dengan inputan yang sudah di pisah dan disimpan di dalam variabel command.Selagi perintah pertama tidak sama dengan exit maka yang akan berjalan adalah sebagai berikut:

1) Jika perintah pertama adalah translate,maka akan program akan memanggil fungsi translate(dengan animasi) dengan 2 parameter yaitu perintah kedua dan ketiga.

2) Jika perintah pertama adalah dilate,maka akan program akan memanggil fungsi dilate(dengan animasi) dengan 1 parameter yaitu perintah kedua.

3) Jika perintah pertama adalah rotate,maka akan program akan memanggil fungsi rotate(dengan animasi) dengan 3 parameter yaitu perintah kedua,ketiga dan keempat.

4) Jika perintah pertama adalah reflect,maka dicek apakah parameter nya x,y,y=x,atau y=-x.Jika iya,program akan memanggil fungsi reflect(dengan animasi) dengan 1 parameter yaitu parameter.Jika bukan,maka program akan memanggil fungsi reflect2(dengan animasi) yang melakukan reflect terhadap (a,b) dengan 1 parameter yaitu parameter.

5) Jika perintah pertama adalah shear,maka program akan memanggil fungsi shear(dengan animasi) dengan 2 parameter yaitu perintah kedua dan ketiga.

6) Jika perintah pertama adalah stretch,maka program akan memanggil fungsi strectch(dengan animasi) dengan 2 parameter yaitu perintah kedua dan ketiga.

7) Jika perintah pertama adalah custom,maka program akan memanggil custom(dengan animasi)

dengan 4 parameter yaitu perintah kedua,ketiga,keempat,kelima.
8) Jika perintah pertama adalah multiple,maka akan melakukan baca masukan perintah sebanyak perintah kedua.Aksi yang dilakukan sama seperti perintah pada nomor 1-7.
9) Jika perintah pertama adalah reset,maka program akan memanggil reset(dengan animasi) untuk mengembalikan ke semula.

Jika perintah pertama salah,maka akan menampilkan pesan kesalahan.
Jika perintah pertama adalah exit,maka akan keluar dari program dan menutup windows GLUT.

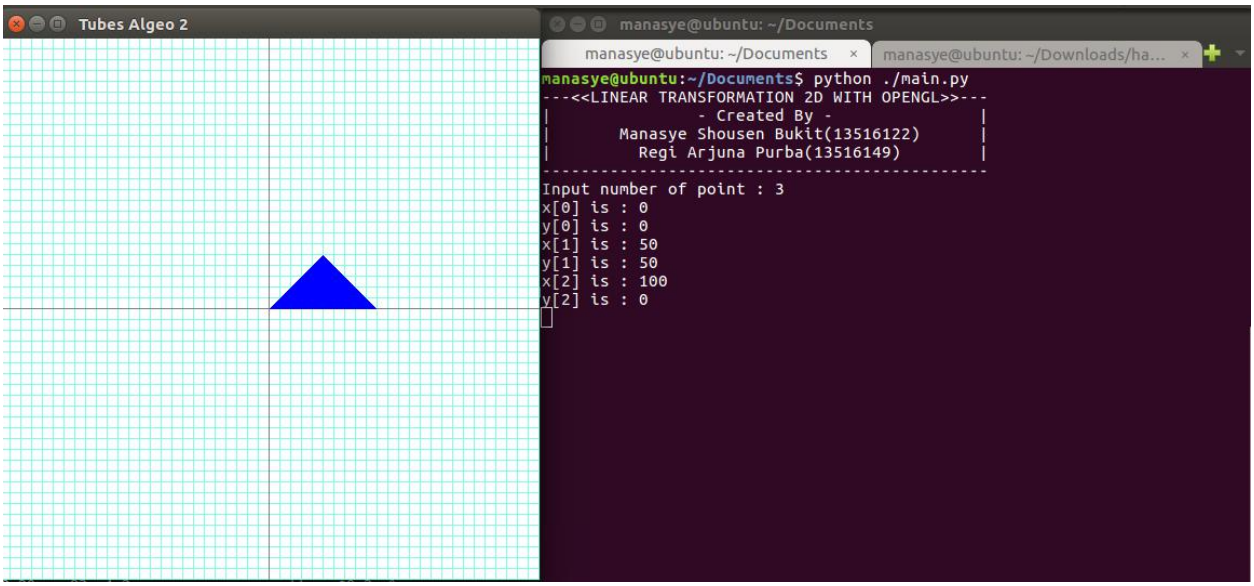
Proses animasi di setiap fungsi serupa.Pertama-tama,kita mempartisi perubahan objek sebesar variable partision.Dengan menggunakan fungsi time.sleep kita dapat seolah-seolah menunda looping partisi sejumlah waktu dan alhasil membuat animasi.

Pembagian Tugas Antar Kelompok :

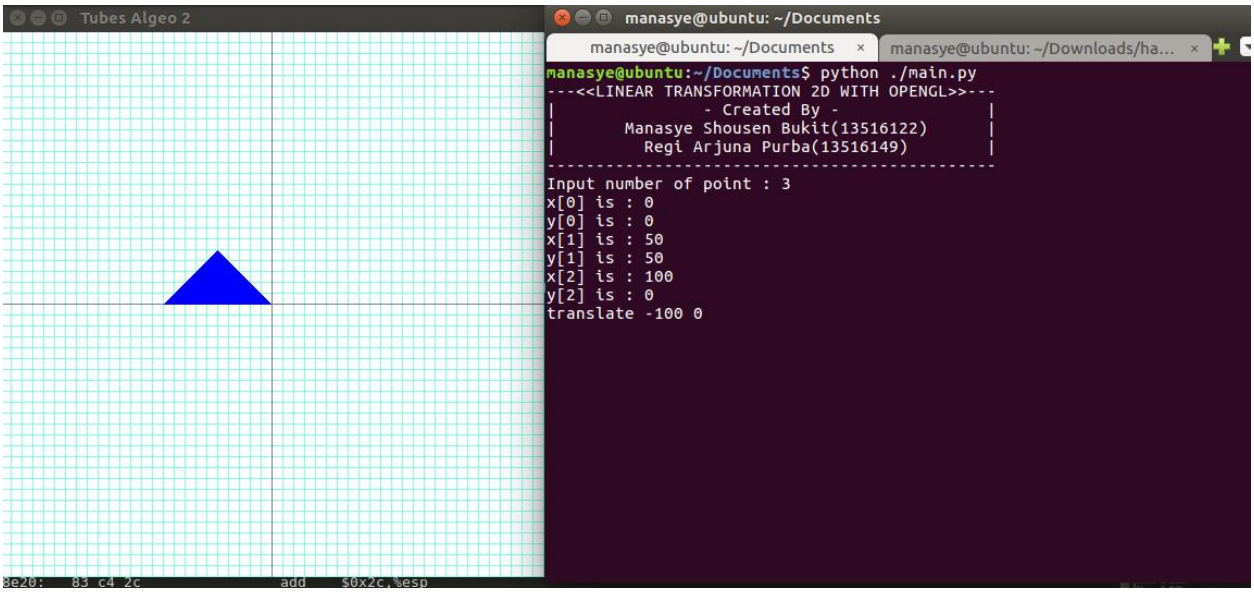
1. Manasye Shousen Bukit (13516122)
   Tugas : Membuat code program dan laporan
2. Regi Arjuna Purba (13516149)
   Tugas : Membuat Laporan

BAB IV

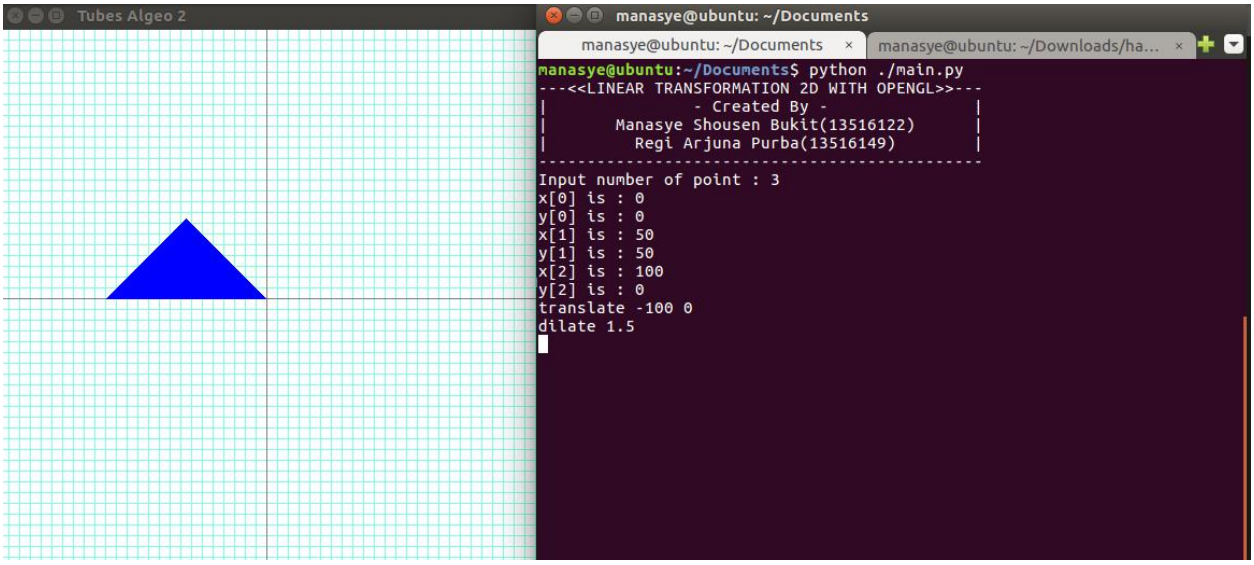
EKSPERIMEN



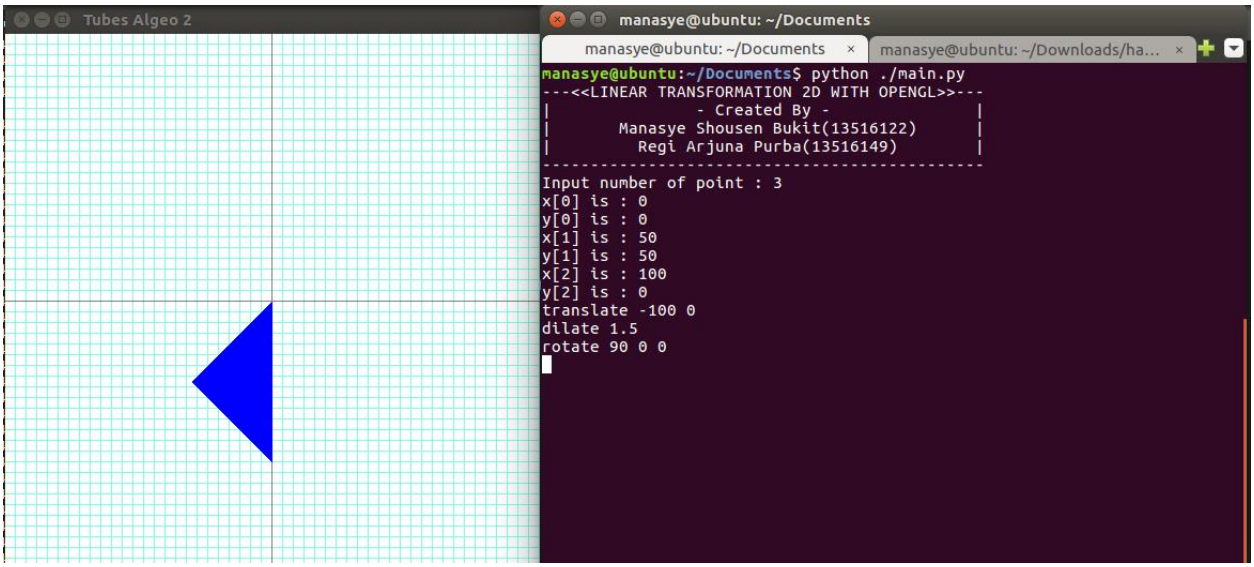
Menerima inputan dari user dan menampilkan objek beserta koordinat ke layar GLUT.



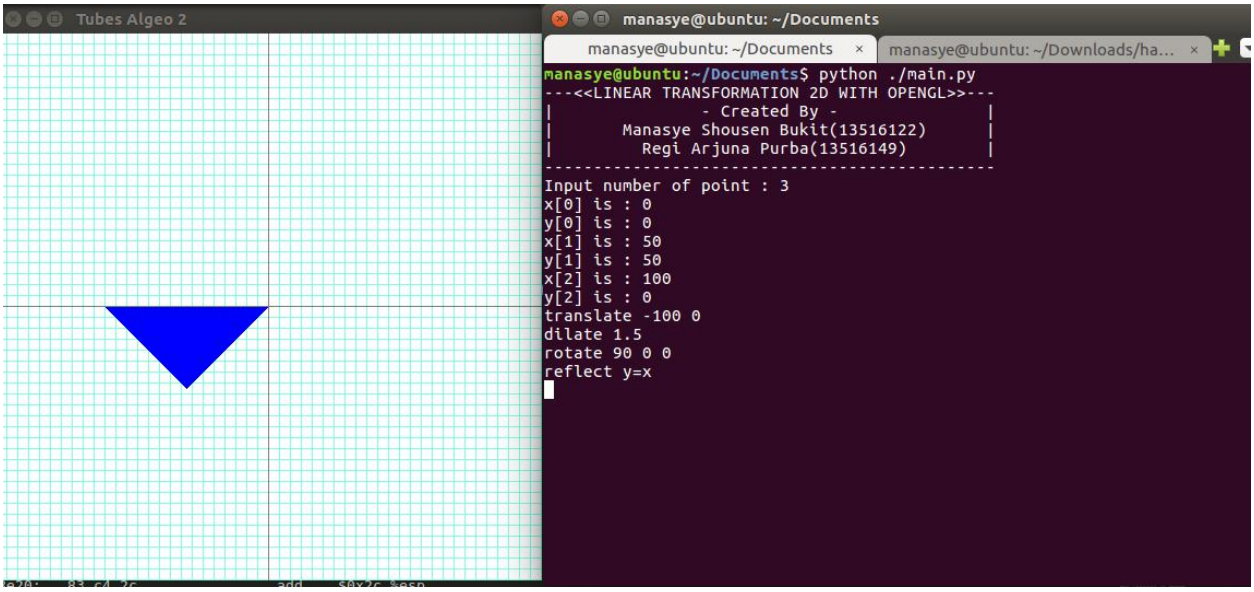
Menerima fungsi translate dan mentranslasikan objek sebesar -100 ke sumbu X dan 0 ke sumbu Y.Proses ini juga memuat animasi perubahan objek.

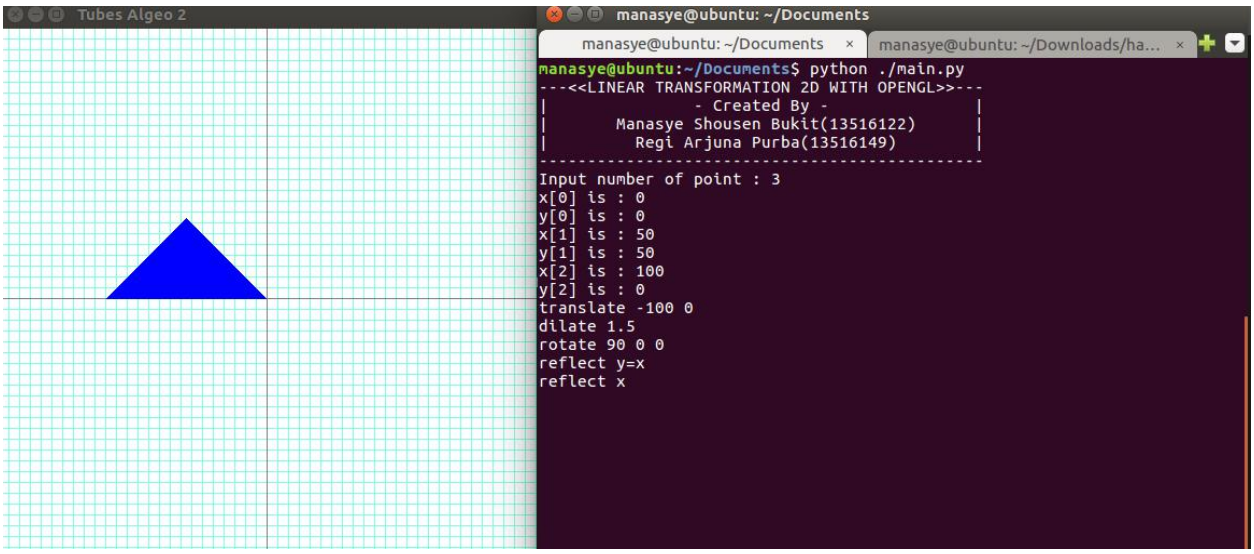


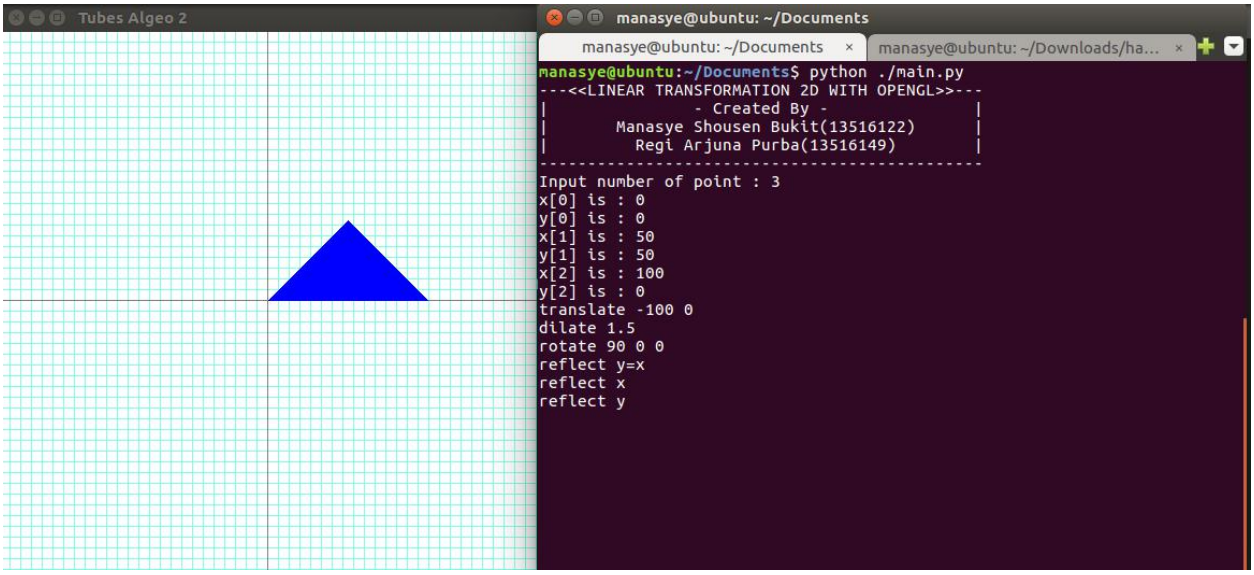
Menerima fungsi dilate dan memperbesar objek sebesar 1.5 kali.Proses ini juga memuat animasi perubahan objek.

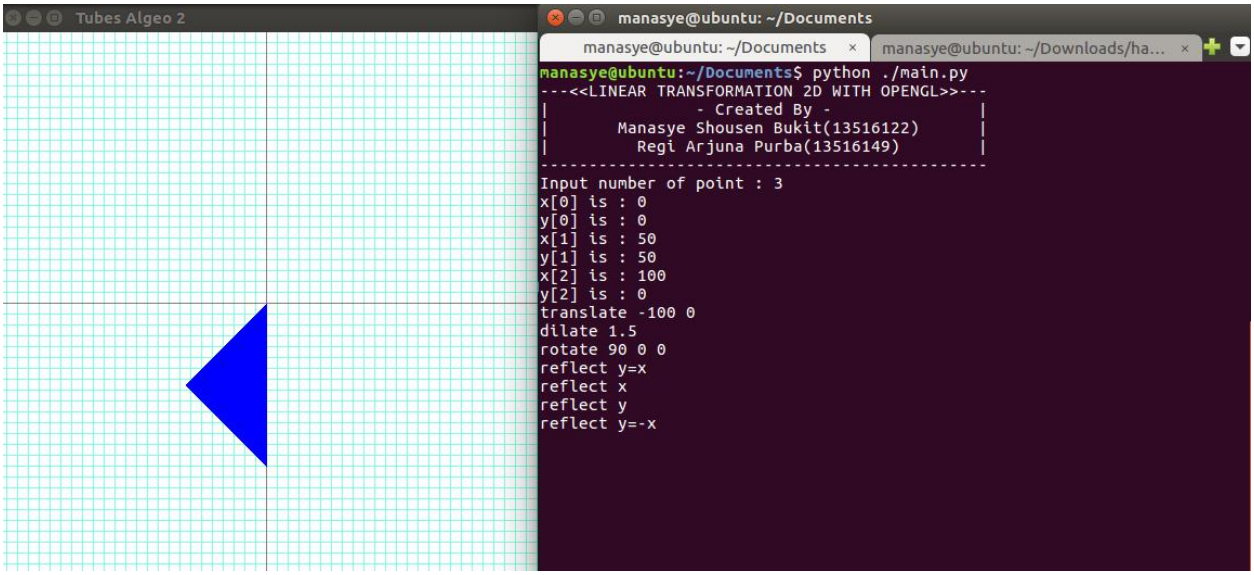


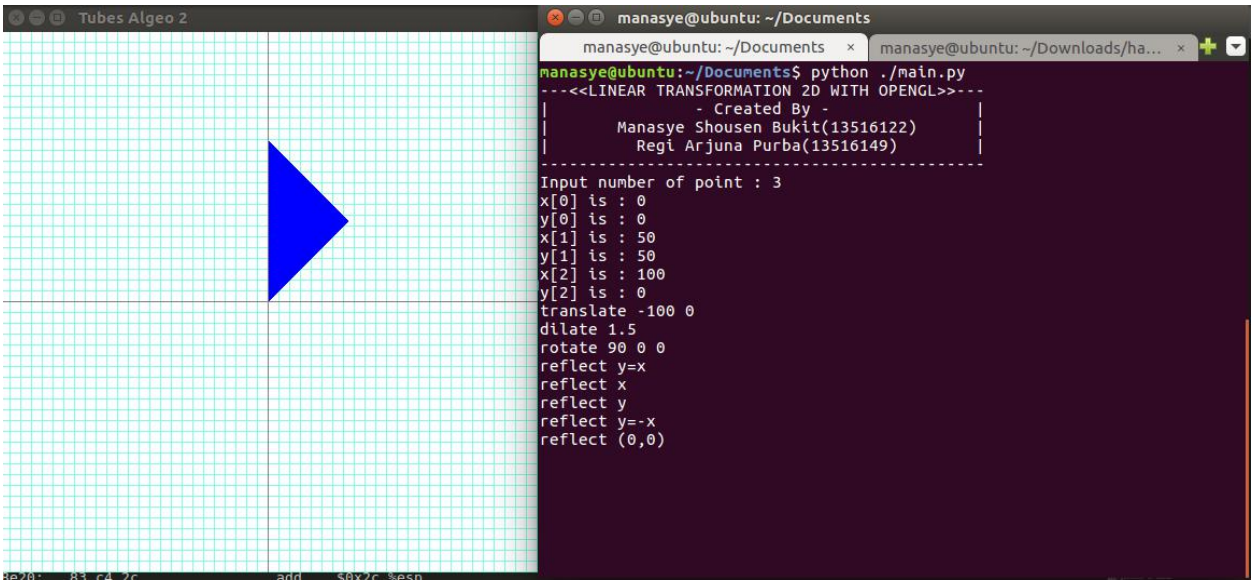
Menerima fungsi rotate dan memutar objek sebesar 90 derajat berlawanan arah jarum jam terhadap titik (0,0).Proses ini juga memuat animasi perubahan objek.


Tubes Algeo 2
manasye@ubuntu: ~/Documents
manasye@ubuntu: ~/Documents
manasye@ubuntu: ~/Downloads/ha...
manasye@ubuntu:~/Documents$ python ./main.py
---<<LINEAR TRANSFORMATION 2D WITH OPENGL>>---
| - Created By - |
| Manasye Shousen Bukit(13516122) |
| Regi Arjuna Purba(13516149) |
Input number of point : 3
x[0] is : 0
y[0] is : 0
x[1] is : 50
y[1] is : 50
x[2] is : 100
y[2] is : 0
translate -100 0
dilate 1.5
rotate 90 0 0
reflect y=x

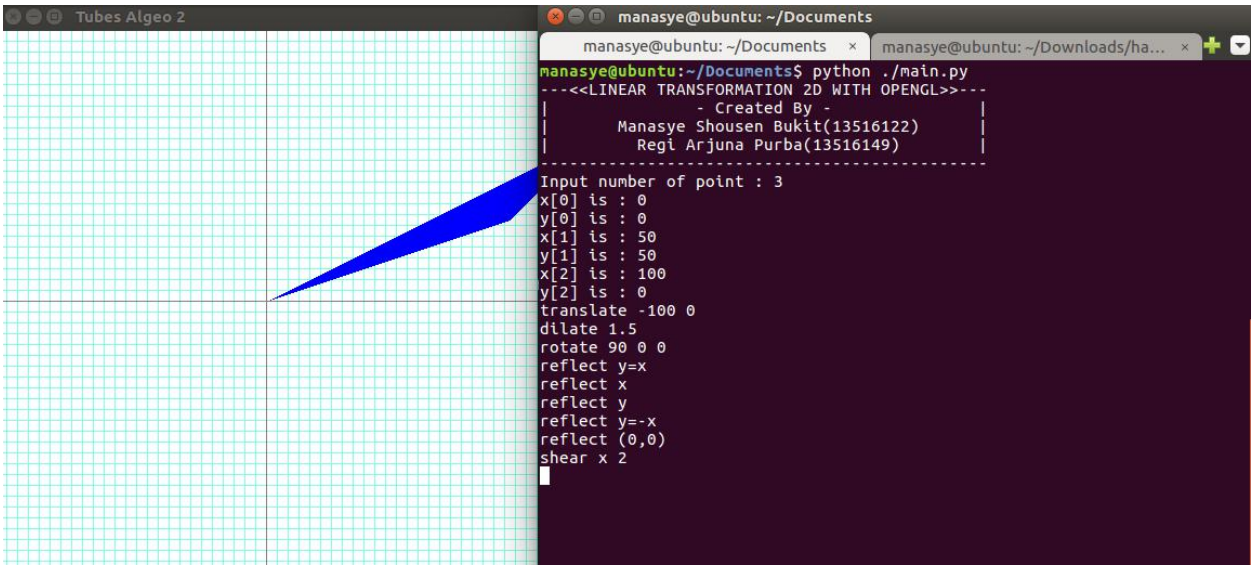

Tubes Algeo 2
manasye@ubuntu: ~/Documents
manasye@ubuntu: ~/Documents
manasye@ubuntu: ~/Downloads/ha...
manasye@ubuntu:~/Documents$ python ./main.py
---<<LINEAR TRANSFORMATION 2D WITH OPENGL>>---
| - Created By - |
| Manasye Shousen Bukit(13516122) |
| Regi Arjuna Purba(13516149) |
Input number of point : 3
x[0] is : 0
y[0] is : 0
x[1] is : 50
y[1] is : 50
x[2] is : 100
y[2] is : 0
translate -100 0
dilate 1.5
rotate 90 0 0
reflect y=x
reflect x


Tubes Algeo 2
manasye@ubuntu: ~/Documents
manasye@ubuntu: ~/Documents
manasye@ubuntu: ~/Downloads/ha...
manasye@ubuntu:~/Documents$ python ./main.py
---<<LINEAR TRANSFORMATION 2D WITH OPENGL>>---
| - Created By - |
| Manasye Shousen Bukit(13516122) |
| Regi Arjuna Purba(13516149) |
Input number of point : 3
x[0] is : 0
y[0] is : 0
x[1] is : 50
y[1] is : 50
x[2] is : 100
y[2] is : 0
translate -100 0
dilate 1.5
rotate 90 0 0
reflect y=x
reflect x
reflect y

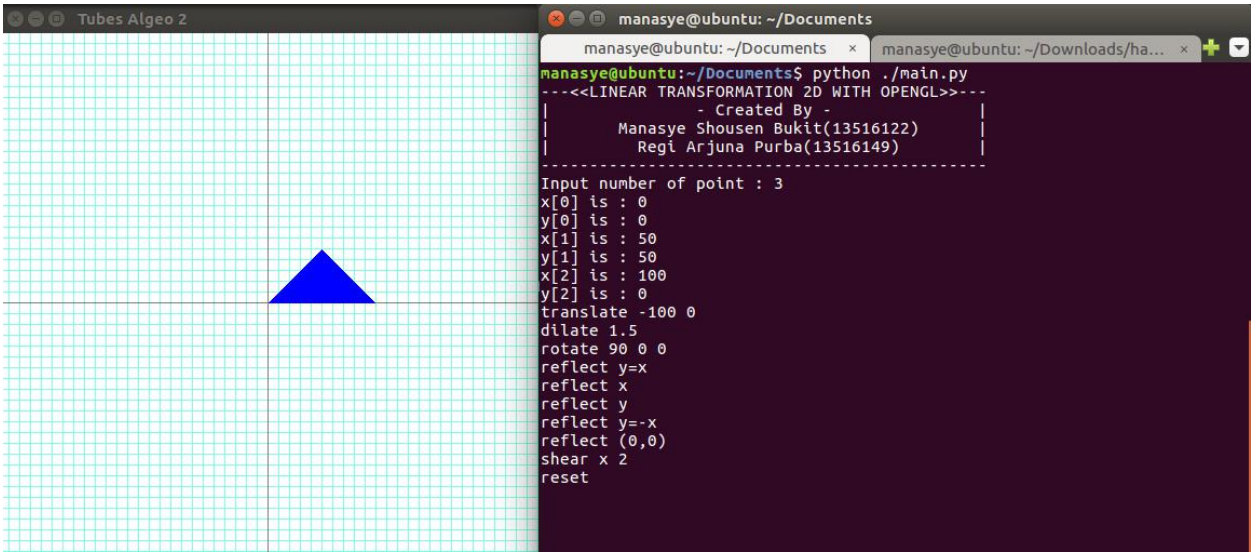




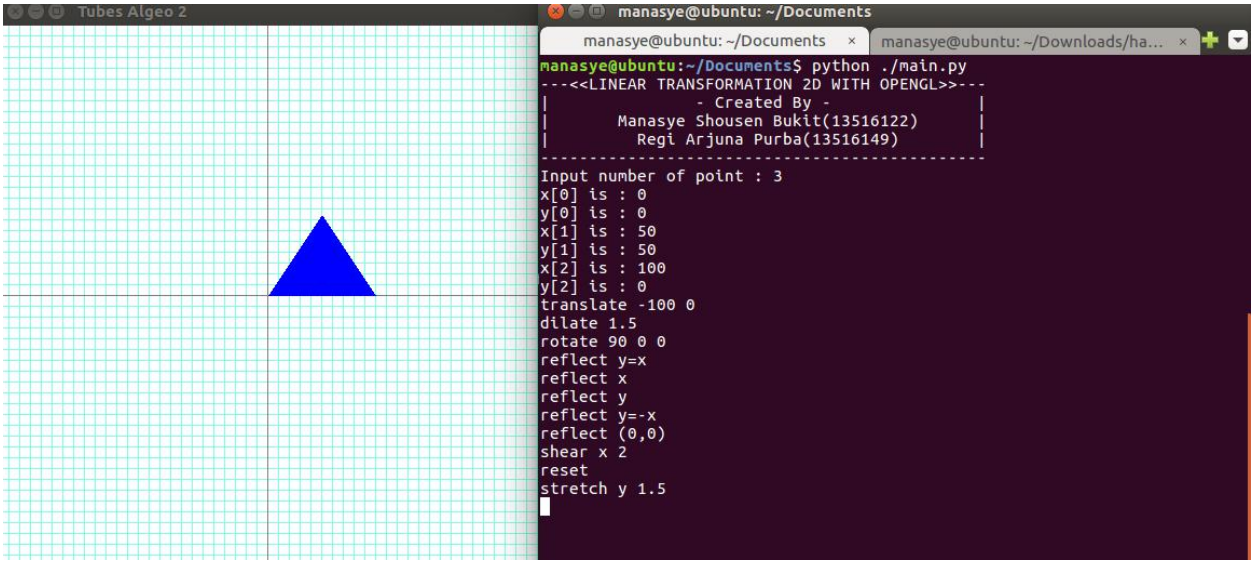
Menerima fungsi reflect dan melakukan pencerminan sesuai parameter masing-masing.Proses ini juga memuat animasi perubahan objek.



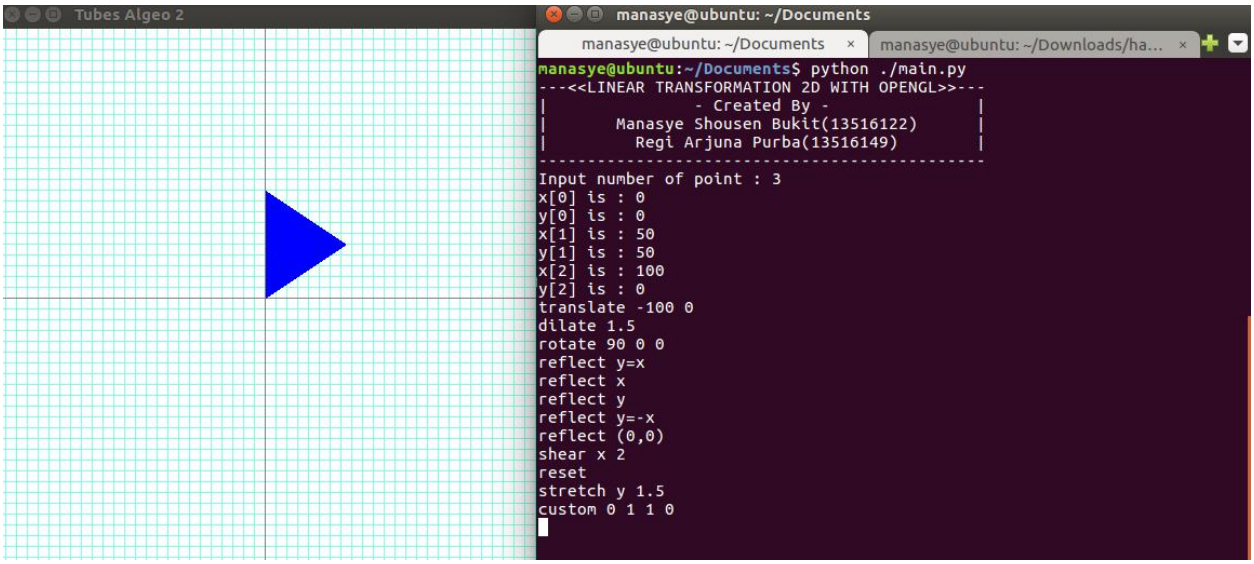
Menerima fungsi shear dan melakukan penggusuran terhadap sumbu x sebesar 2 kali.Proses ini juga memuat animasi perubahan objek.



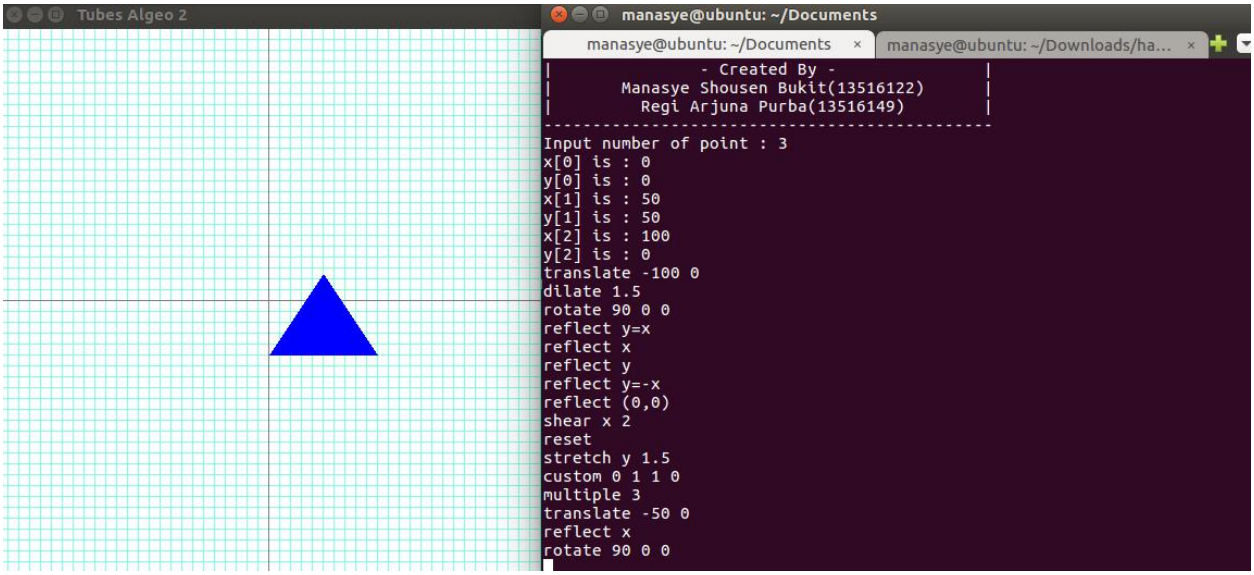
Mengembalikan ke koordinat semula.



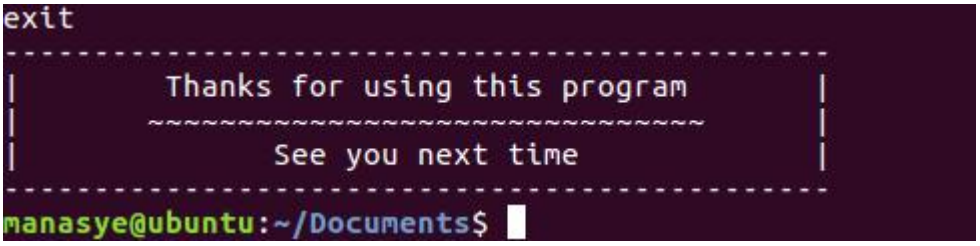
Menerima fungsi stretch dan melakukan peregangan terhadap sumbu y sebesar 1.5 kali.Proses ini juga memuat animasi perubahan objek.



Menerima fungsi custom dan melakukan transformasi dengan matriks transformasi [0,1,1,0].Proses ini juga memuat animasi perubahan objek.



Melakukan multiple fungsi sebanyak 3 kali,yaitu geser objek searah sumbu X negatif sebanyak 50,pencerminan terhadap sumbu X,dan rotasi 90 derajat terhadap titik (0,0).Proses ini juga memuat animasi perubahan objek.



Exit akan menghentikan program dan menutup layar GLUT.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Hasil yang dicapai dari program yang dibuat adalah program dapat melakukan perhitungan matematis pada operasi transformasi linier dan menggambarkannya dalam layar untuk menunjukkan proses transformasi dari asal menjadi bayangannya. Program akan bekerja sesuai input perintah dari pengguna. Input translate untuk mengubah posisi objek dari titik asal ke titik bayangan, input rotate untuk merotasi objek dan input-input lain yang tersedia pada program. Program memanfaatkaan matriks transformasi dalam perhitungan. Dengan memanfaatkan matriks transformasi, operasi matematis lebih mudah dan sederhana.

5.2 Saran

Program yang kami buat belum dapat melakukan transformasi linier pada objek 3 dimensi, hanya objek 2 dimensi. Pengembangan selanjutnya akan membuat program dapat mengolah objek-objek 3 dimensi.Input masih dilakukan pada terminal atau command prompt. Pengembangan lebih lanjut bisa dilakukan dengan membuat GUI sendiri untuk input perintah. Ditemukan beberapa kesulitan dalam menggunakan library OpenGl untuk menyelesaikan program ini, karena masih baru pertama sekali digunakan.

Sumber :

https://mathspace.co/accounts/login/?next=/learn/world-of-maths/functions-graphs-and-behaviour/matrix-transformations-of-functions-52385/matrix-transformations-of-functions-1904/
https://www.varsitytutors.com/hotmath/hotmath_help/topics/transformation-of-graphs-using-matrices-dilation
http://www.konsep-matematika.com/2017/02/komposisi-transformasi-dengan-matriks.html
http://repository.binus.ac.id/content/K0034/K003481224.pdf