TECHNICAL REPORT PRAKTIKUM
GRAFIKA KOMPUTER MODUL 4
"TRANSFORMASI GEOMETRI"

Disusun Oleh :

TGL PRAKTIKUM	: 9 APRIL 2018
NAMA	: REISKI MUHAMMAD RA
NRP	: 160411100030
KELAS	: GRAFKOM E
DOSEN PENGAMPU	: FITRIYATUL QOMARIYAH, S.Kom., M.Kom.
ASISTEN	: SOFIAN EKA SANDRA

Disetujui : ...../……./……../Bangkalan
(SOFIAN EKA SANDRA**)** 150411100108

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO MADURA

2017/2018

BAB I

PENDAHULUAN

1.1 DASAR TEORI

Dalam matematika, transformasi adalah fungsi yang memetakan suatu set X ke set yang lain ataupun ke set X sendiri. Dalam dunia Grafika Komputer, set X (yang mengalami proses transformasi) biasanya berupa strukur geometri, sehingga disebut transformasi geometri. Terdapat banyak jenis operasi transformasi geometri: translasi, refleksi, rotasi, scaling, shearing.

OpenGL memiliki 3 perintah transformasi:

a.) glTranslated(a, b, c): melakukan operasi translasi/pergeseran sejauh a pada sumbu x, sejauh b pada sumbu y, dan sejauh c pada sumbu z. Contoh: jika ingin menggeser obyek sejauh 4 pada sumbu x dan -3 pada sumbu y, maka perintahnya adalah: glTranslated(4.0, -3.0, 0.0).

b.) glScaled(d, e, f): melakukan penskalaan sebesar d pada sumbu x, sebesar e pada sumbu y, sebesar f pada sumbu z. Contoh: jika ingin memperbesar obyek pada sumbu x sebesar 2 kali dan memperkecil obyek menjadi seperempatnya, maka perintahnya adalah: glScaled(2.0, 0.25, 0.0).

c.) glRotated(alpha, i, j, k): melakukan rotasi sebesar alpha. Alpha ada dalam satuan derajat, bukan radian. i, j, dan k mewakili sumbu rotasi x, y, dan z. Set nilainya menjadi 1.0 pada sumbu yang diingikan. Contoh: jika ingin merotasi obyek sebesar 90 derajat pada sumbu x, maka perintahnya adalah: glRotated(90.0,
1, 0, 0).

*huruf d diakhir perintah transformasi merupakan kependekan dari double; yang berarti argumen a, b, c, d, e, f, alpha, i, j, dan k adalah angka pecahan presisi ganda (double). Selain double (d), pilihan jenis argumen yang dapat digunakan adalah: i(integer/bilangan bulat) dan f(float/bilangan pecahan presisi tunggal), contoh: glTranslatef(m, n, o). Proses transformasi di OpenGL bersifat melekat: sekali sebuah perintah transformasi dieksekusi, perintah tersebut akan selalu dilakukan untuk semua perintah yang ada dibawahnya. Contoh: jika pada program terdapat perintah glTranslated(10.0, 0.0, 0.0) pada baris ke 25, maka perintah- perintah glVertex pada baris ke 26 dan seterusnya akan selalu ditranslasi pada sumbu x sejauh 10. Program di bawah ini menunjukkan contoh translasi.

Contoh Program Translasi

```
void display()
{
 glClear(GL_COLOR_BUFFER_BIT);
 glColor3f(0.0,0.0,0.5);
 // Gambar kotak pertama di sudut kiri bawah
 glRecti(0,0, 10, 10);
 //translasi ke 20, 20
 glTranslated(20.0, 20.0, 0);
 glRecti(0,0, 10, 10);
```

```
 glFlush();
}
void myinit()
{
 glMatrixMode(GL_PROJECTION);
 glLoadIdentity();
 gluOrtho2D(0.0,50.0,0.0,50.0);
 glMatrixMode(GL_MODELVIEW);
 glClearColor(1.0,1.0,1.0,1.0);
 glColor3f(0.0,0.0,0.0);
}
int main(int argc, char* argv[])
{
 glutInit(&argc,argv);
 glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
 glutInitWindowSize(400,400);
 glutInitWindowPosition(100,100);
 glutCreateWindow("Transform");
 glutDisplayFunc(display);
 myinit();
 glutMainLoop();
 return 0;
}
```

Contoh Program Scaling

```
void display()
{
 glClear(GL_COLOR_BUFFER_BIT);
 glColor3f(0.0,0.0,0.5);
 // Gambar kotak pertama di sudut kiri bawah
 glRecti(0,0, 10, 10);
 //Scaling kotak yang digambar di ke 20, 20 sebesar 1.5 kali
 glScaled(1.5, 1.5, 0.0);
 glRecti(20,20, 30, 30);
 glFlush();
}
void myinit()
{
 glMatrixMode(GL_PROJECTION);
 glLoadIdentity();
 gluOrtho2D(0.0,50.0,0.0,50.0);
 glMatrixMode(GL_MODELVIEW);
 glClearColor(1.0,1.0,1.0,1.0);
glColor3f(0.0,0.0,0.0);
}
int main(int argc, char* argv[])
{
 glutInit(&argc,argv);
 glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
 glutInitWindowSize(400,400);
```

```
 glutInitWindowPosition(100,100);
 glutCreateWindow("Transform");
 glutDisplayFunc(display);
 myinit();
 glutMainLoop();
 return 0;
}
```

Yang perlu diperhatikan disini adalah bahwa proses scaling dilakukan dari sumbu koordinat yang terletak di sudut kiri bawah jendela. Hal inilah yang menyebabkan tampilan pada program 4.2 diatas terlihat cenderung lebih ke kanan atas jendela.

Contoh Program Rotasi

```
void display()
{
 glClear(GL_COLOR_BUFFER_BIT);
 glColor3f(0.0,0.0,0.5);
 // Gambar kotak pertama di sudut kiri bawah
 glRecti(0,0, 10, 10);
 //rotasi kotak kedua sebesar 15 derajat terhadap sumbu
koordinat(titik kiri bawah)
 glRotated(15, 0, 0, 1.0);
 glRecti(20,20, 30, 30);
 glFlush();
}
void myinit()
{
 glMatrixMode(GL_PROJECTION);
 glLoadIdentity();
 gluOrtho2D(0.0,50.0,0.0,50.0);
 glMatrixMode(GL_MODELVIEW);
 glClearColor(1.0,1.0,1.0,1.0);
 glColor3f(0.0,0.0,0.0);
}
int main(int argc, char* argv[])
{
 glutInit(&argc,argv);
 glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
 glutInitWindowSize(400,400);
 glutInitWindowPosition(100,100);
 glutCreateWindow("Transform");
 glutDisplayFunc(display);
 myinit();
 glutMainLoop();
 return 0;
}
```

Yang perlu diperhatikan dari program di atas adalah bahwa rotasi dilakukan terhadap titik koordinat yang terletak pada ujung kiri bawah jendela. Supaya rotasi terjadi pada titik tengah obyek, perlu dilakukan kombinasi perintah transformasi.

Kombinasi transformasi

Operasi-operasi transformasi yang berbeda dapat dikombinasikan. Contoh: jika ingin melakukan operasi-operasi berikut pada sebuah obyek:

translasi sebesar (3, 4)

lalu rotasi sebesar 30° pada sumbu z

lalu skala sebesar (1.5, 1.5)

lalu translasi lagi sebesar (0, 0.5)

dan terakhir rotasi sebesar 45°

maka perintah-perintahnya adalah:

```
glRotated(45, 0, 0, 1);
glTranslated(0.0, 0.5, 0.0);
glScaled(1.5, 1.5, 0.0);
glRotated(30.0, 0, 0, 1);
glTranslated(3.0, 4.0, 0.0);
```

Yang perlu diperhatikan disini adalah urutan perintah. OpenGL melakukan perintah transformasi mulai dari yang paling bawah.

Perlu diingat pula bahwa karena pada dasarnya operasi transformasi dilakukan dengan menggunakan operasi perkalian matrix yang tidak bersifat komutatif ($AB \neq BA$), maka urutan operasi transformasi sangat berpengaruh. Program di bawah menunjukan contoh kombinasi transformasi

```
void display()
{
 glClear(GL_COLOR_BUFFER_BIT);
 glColor3f(0.0,0.0,0.5);
 // Gambar kotak pertama di kuadran 1
 glRecti(0,0, 2, 2);
 // Gambar kotak kedua
 glRotated(45, 0, 0, 1);
 glTranslated(0.0, 0.5, 0.0);
```

```
 glScaled(1.5, 1.5, 0.0);
 glRotated(30.0, 0, 0, 1);
 glTranslated(3.0, 4.0, 0.0);
 glRecti(0,0, 2, 2);
 glLoadIdentity();
 //Gambar sumbu koordinat
 glColor3f(1.0, 1.0, 0.0);
 glBegin(GL_LINES);
 glVertex3f(-10.0,0.0,0.0);
glVertex3f(10.0,0.0,0.0);
 glVertex3f(0.0,-10.0,0.0);
 glVertex3f(0.0,10.0,0.0);
 glEnd();
 glFlush();
}
void myinit()
{
 glMatrixMode(GL_PROJECTION);
 glLoadIdentity();
 gluOrtho2D(-10.0, 10.0, -10.0, 10.0);
 glMatrixMode(GL_MODELVIEW);
 glClearColor(1.0,1.0,1.0,1.0);
 glColor3f(0.0,0.0,0.0);
}
int main(int argc, char* argv[])
{
 glutInit(&argc,argv);
 glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
 glutInitWindowSize(400,400);
 glutInitWindowPosition(100,100);
 glutCreateWindow("Transform");
 glutDisplayFunc(display);
 myinit();
 glutMainLoop();
 return 0;
}
```

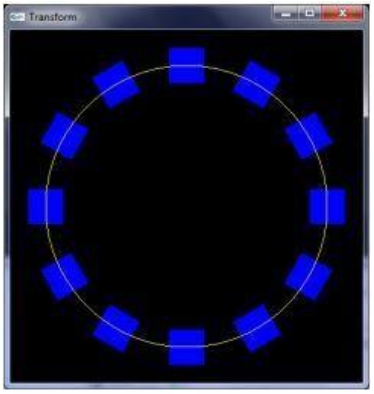
Program di atas memiliki baris perintah glLoadIdentity() sebelum menggambar sumbu koordinat. Hal ini disebabkan, tiap kali perintah transformasi diberikan semua baris perintah vertex di bawahnya akan terpengaruh. Perintah glLoadIdentity() digunakan untuk menetralisir efek ini; sehingga, pemanggilan perintah vertex dibawahnya akan kembali normal seperti sebelum terjadi transformasi.

BAB II
TUGAS DAN IMPLEMENTASI

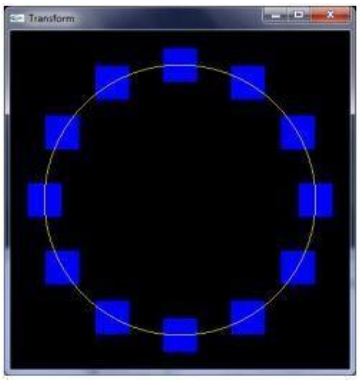
2.1 TUGAS

1. Buat checker board (papan catur) miring sebagai berikut dengan menggunakan Transformasi!

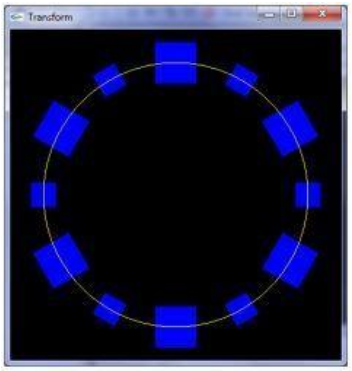
2. Buat kotak berputar berikut menggunakan transformasi!



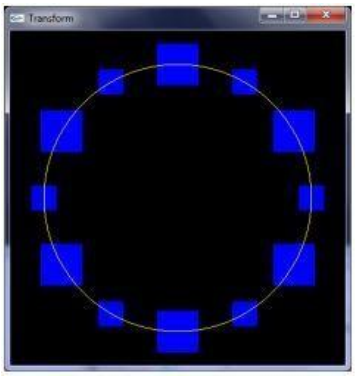
3. Buat kotak berputar berikut menggunakan transformasi



4. Buat variasi kotak berputar berikut menggunakan transformasi



5. Buat variasi kotak berputar berikut menggunakan transformasi



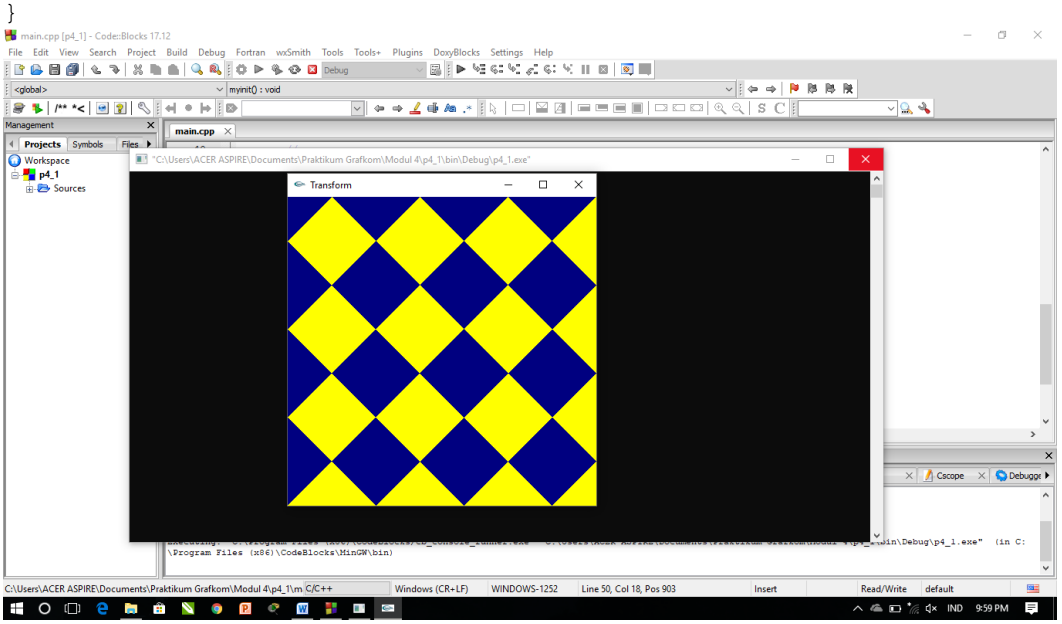
2.2 PENYELESAIAN

1. Program 1

```
#include <windows.h>
#ifdef __APPLE__
#include <GLUT/glut.h>
#else
#include <GL/glut.h>
#endif
//Supaya bisa menggunakan fungsi sin(), program perlu include
Math.h
#include <Math.h>
void display()
{
glClear(GL_COLOR_BUFFER_BIT);
```

```
glRotated(45, 0, 0, 1);
for (int t=0;t<=50;t+=3){
    glTranslated(-160,10,0);
    int a = 1;
    for (int i=0;i<=50;i+=3){
        if(a%2==0){
            glColor3f(0.0,0.0,0.5);
            glTranslated(10,0,0);
            glRecti(30,-60, 40, -50);
        }
        else{
            glColor3f(1.0,1.0,0);
            glTranslated(10,0,0);
            glRecti(30,-60, 40, -50);
        }
        a++;
    }
}
glFlush();
}
void myinit()
{ glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0,49.5,0,49.5);
glMatrixMode(GL_MODELVIEW);
glClearColor(1.0,1.0,1.0,1.0);
glColor3f(0.0,0.0,0.0);
}
int main(int argc, char* argv[])
{ glutInit(&argc,argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400,400);
glutInitWindowPosition(100,100);
```

```
glutCreateWindow("Transform");
glutDisplayFunc(display);
myinit();
glutMainLoop();
return 0;
}
```

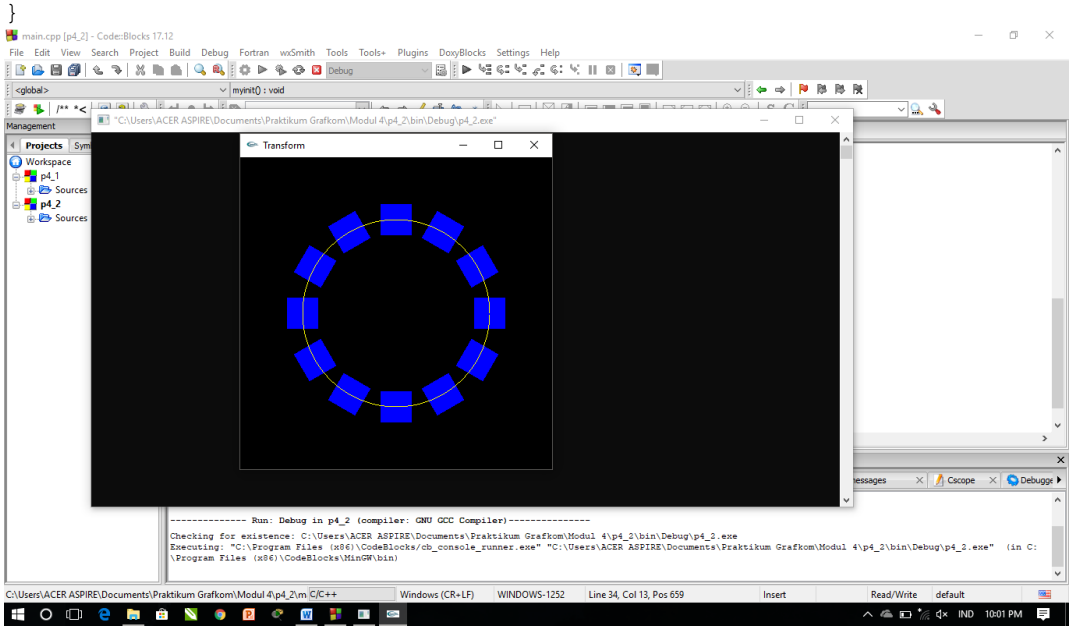


2. Program 2

```
#include <windows.h>

#ifdef __APPLE__
#include <GLUT/glut.h>
#else
#include <GL/glut.h>
#endif
#include<math.h>

void lingkaran(){
    glColor3f(0.0,0.0,1.0);
    for (int i=1; i<=12; i++){
        glRectf(-1, 5, 1, 7);
        glRotated(30, 0, 0, 1);
    }
    glColor3f(1.0,1.0,0.0);
    glBegin(GL_LINE_STRIP);
    for (float i = 0.0; i<=6.28; i+=0.01){
        glVertex2f(6*cos(i), 6*sin(i));
    }
```

```
    glEnd();
}
void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    lingkaran();
    glFlush();
}
```

```
void myinit()
{
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-10.0,10.0,-10.0,10.0);
    glMatrixMode(GL_MODELVIEW);
    glClearColor(0.0,0.0,0.0,1.0);
    glColor3f(0.0,0.0,0.0);
}
int main(int argc, char* argv[])
{ glutInit(&argc,argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400,400);
glutInitWindowPosition(100,100);
glutCreateWindow("Transform");
glutDisplayFunc(display);
myinit();
glutMainLoop();
return 0;
}
```



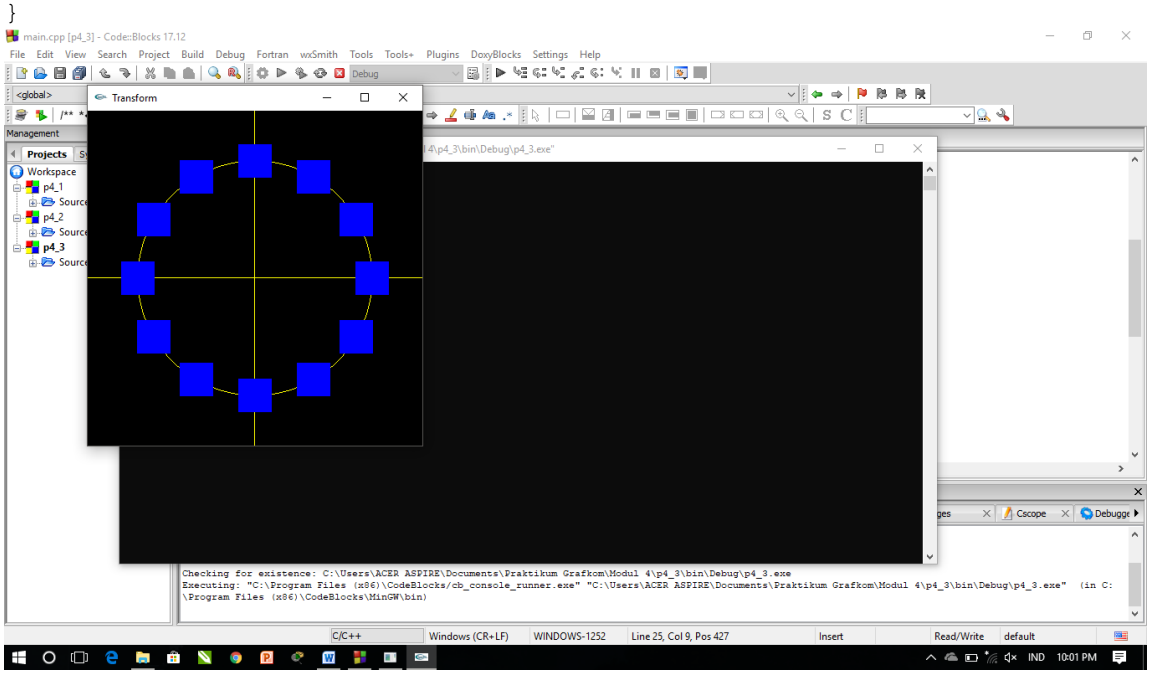
3. Program 3

```
#include <windows.h>
#ifdef __APPLE__
#include <GLUT/glut.h>
#else
#include <GL/glut.h>
#endif
```

```
#include <stdlib.h>
#include <math.h>

void display()
{
glClear(GL_COLOR_BUFFER_BIT);
```

```
glColor3f(1.0, 1.0, 0.0);
// Gambar lingkaran
glBegin(GL_LINE_LOOP);
    for(float t=0.0; t<=6.28; t+=0.01)
    {
        glVertex2f(7*cos(t),7*sin(t));
    }
glEnd();
//Gambar sumbu koordinat

glBegin(GL_LINES);
glVertex3f(-10.0,0.0,0.0);
glVertex3f(10.0,0.0,0.0);
glVertex3f(0.0,-10.0,0.0);
glVertex3f(0.0,10.0,0.0);
glEnd();

glColor3f(0.0,0.0,1);
// Gambar kotak
for(float i=0;i<=6.28;i+=6.28/12){
glTranslated(7*sin(i),7*cos(i),0);
glRecti(-1,-1, 1, 1);
glLoadIdentity();
}
glFlush();
}
void myinit()
{
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(-10.0, 10.0, -10.0, 10.0);
glMatrixMode(GL_MODELVIEW);
glClearColor(0.0,0.0,0.0,0.0);
glColor3f(0.0,0.0,0.0);
```

```
}
int main(int argc, char* argv[])
{ glutInit(&argc,argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400,400);
glutInitWindowPosition(100,100);
glutCreateWindow("Transform");
glutDisplayFunc(display);
myinit();
glutMainLoop();
return 0;
}
```



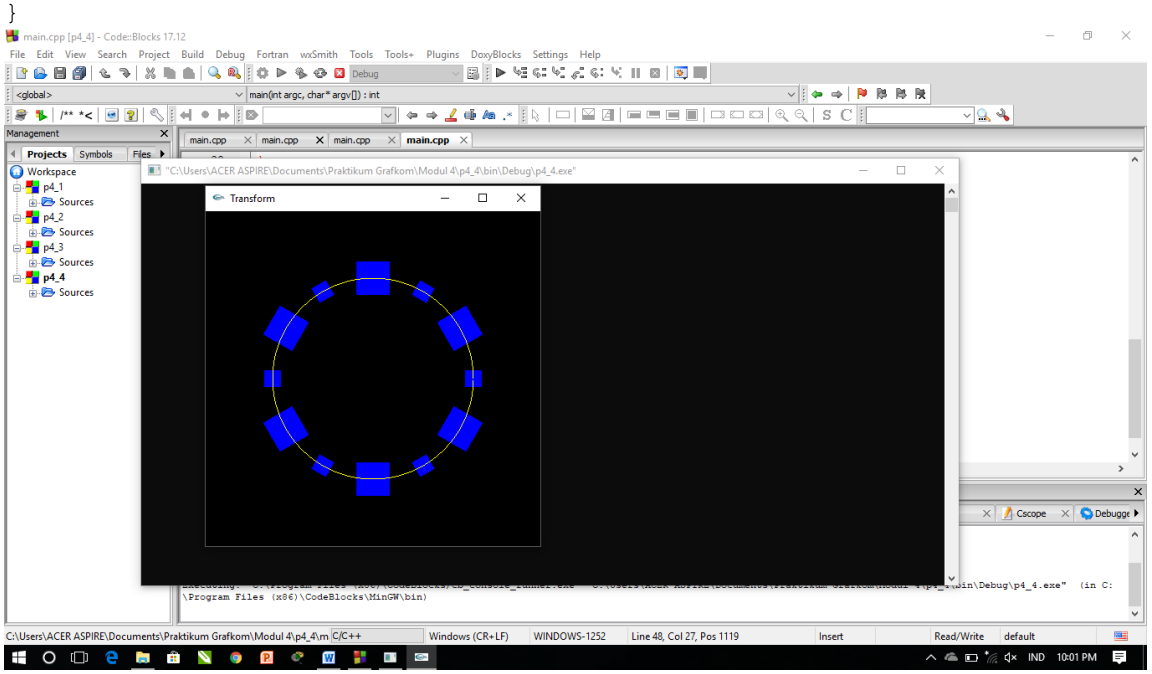
4. Program 4

```
#include <windows.h>

#ifdef __APPLE__
#include <GLUT/glut.h>
#else
#include <GL/glut.h>
#endif
#include<math.h>

void lingkaran(){
    glColor3f(0.0,0.0,1.0);
    int sudut = 0;
```

```
for (int i=1; i<=12; i++){ //Perulangan for sampe 12
    if(i%2!=0){ //Jika i modulus 2 tidak sama dengan 0
        glRotated(sudut, 0, 0, 1);
        glRectf(-1, 5, 1, 7); //Kotak
        glLoadIdentity(); //Normalisasi
    }else{
        glRotated(sudut, 0, 0, 1);
        glRectf(-0.5, 5.5, 0.5, 6.5);
```

```
            glLoadIdentity();
        }
        sudut+=30;
    }
    glColor3f(1.0,1.0,0.0);
    glBegin(GL_LINE_STRIP);
    for (float i = 0.0; i<=6.28; i+=0.01){ //Lingkaran
        glVertex2f(6*cos(i), 6*sin(i));
    }
    glEnd();
}
void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    lingkaran();
    glFlush();
}
void myinit()
{
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-10.0,10.0,-10.0,10.0);
    glMatrixMode(GL_MODELVIEW);
    glClearColor(0.0,0.0,0.0,1.0);
    glColor3f(0.0,0.0,0.0);
}
int main(int argc, char* argv[])
{ glutInit(&argc,argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400,400);
glutInitWindowPosition(100,100);
glutCreateWindow("Transform");
glutDisplayFunc(display);
myinit();
```

```
glutMainLoop();
return 0;
}
```

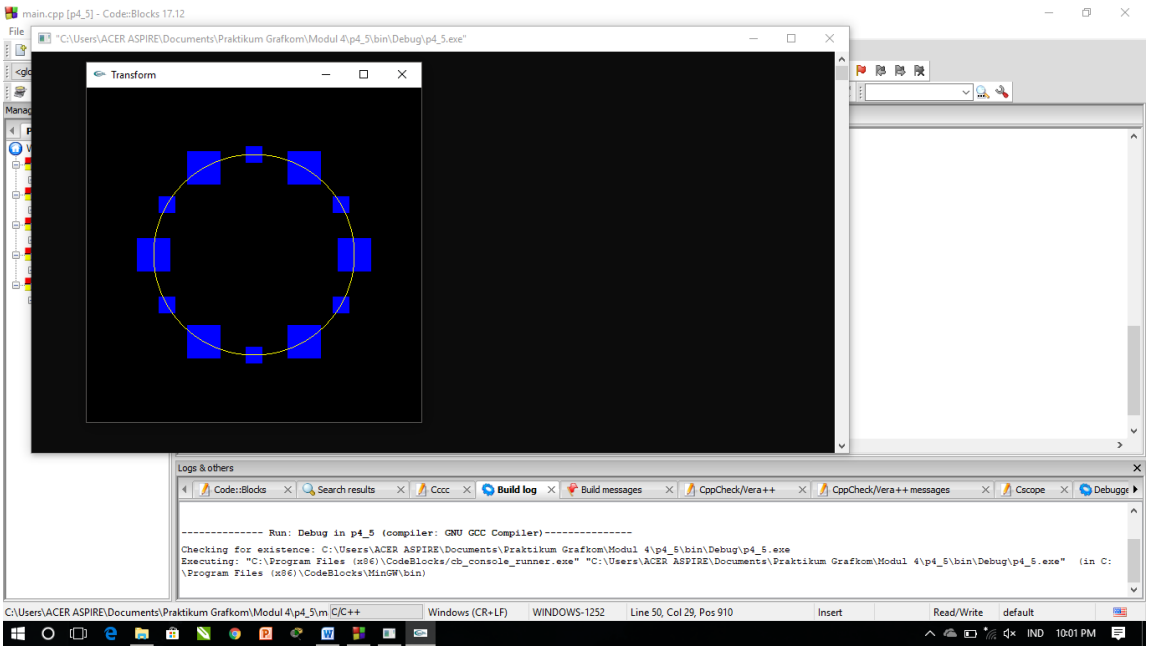


5. Program 5

```
#include <windows.h>


#ifdef __APPLE__
#include <GLUT/glut.h>
#else
#include <GL/glut.h>
#endif
#include<math.h>
void kotak (){
    glColor3f(0,0,1.0);
   int j=0;
for(float i=0;i<=6.28;i+=6.28/12){
if(j%2==0){ glScaled(0.5,0.5,0);
    glTranslated(12*sin(i),12*cos(i),0);
    glRecti(-1,-1, 1, 1);
    glLoadIdentity();
}else{
    glTranslated(6*sin(i),6*cos(i),0);
    glRecti(-1,-1, 1, 1);
    glLoadIdentity();
}
j+=1;
```

```
}
}
void lingkaran(){

    glColor3f(1.0, 1.0, 0.0);
// Gambar lingkaran
glBegin(GL_LINE_LOOP);
```

```
    for(float t=0.0; t<=6.28; t+=0.01)
    {
        glVertex2f(6*cos(t),6*sin(t));
    }
glEnd();
}
void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    kotak();
    lingkaran();

    glFlush();
}
void myinit()
{
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-10.0,10.0,-10.0,10.0);
    glMatrixMode(GL_MODELVIEW);
    glClearColor(0.0,0.0,0.0,1.0);
    glColor3f(0.0,0.0,0.0);
}
int main(int argc, char* argv[])
{ glutInit(&argc,argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(400,400);
glutInitWindowPosition(100,100);
glutCreateWindow("Transform");
glutDisplayFunc(display);
myinit();
glutMainLoop();
return 0;
}
```


main.cpp [p4_5] - Code::Blocks 17.12
"C:\Users\ACER ASPIRE\Documents\Praktikum Grafkom\Modul 4\p4_5\bin\Debug\p4_5.exe"
Transform
Logs & others
Code::Blocks
Search results
Cccc
Build log
Build messages
CppCheck/Vera++
CppCheck/Vera++ messages
Cscope
Debugge
---------- Run: Debug in p4_5 (compiler: GNU GCC Compiler)----------
Checking for existence: C:\Users\ACER ASPIRE\Documents\Praktikum Grafkom\Modul 4\p4_5\bin\Debug\p4_5.exe
Executing: "C:\Program Files (x86)\CodeBlocks/cb_console_runner.exe" "C:\Users\ACER ASPIRE\Documents\Praktikum Grafkom\Modul 4\p4_5\bin\Debug\p4_5.exe" (in C:
\Program Files (x86)\CodeBlocks\MinGW\bin)
C/C++
Windows (CR+LF)
WINDOWS-1252
Line 50, Col 29, Pos 910
Insert
Read/Write
default
IND
10:01 PM

2.3 PENJELASAN PROGRAM

Pada openGl tak hanya ada fungsi-fungsi untuk membuat suatu bentuk, tapi ada pula fungsi-fungsi untuk mengatur tampilan bentuk-bentuk yang telah dibuat, salah satu contohnya adalah yang dibahas pada modul ini yaitu fungsi- fungsi transformasi, yang terdiri dari translasi (memindahkan posisi objek), scaling (merubah ukuran objek), rotasi (merubah posisi objek), dan kombinasi dari ketiganya. Program-program di atas merupakan contoh perogram yang menerapkan fungsi-fungsi tersebut, di mana pada program pertama dan ke-2 menerapkan fungsi translasi dan rotasi, pada program ke-3 hanya menerapkan fungsi translasi saja, sedangkan pada program ke-4 dan ke-5 menerapkan kombinasi antara fungsi translasi, scaling, dan rotasi.

BAB III
PENUTUP

3.1 KESIMPULAN

Pada OpenGL kita dapat membuat beberapa bentuk bangun datar 2D dengan menentukan titik-titik koordinat nya terlebih dahulu dan juga menggunakan fungsi-fungsi matematika. Dalam membuat bentuk-bentuk menggunakan perhitungan matematika tentunya selain kemampuan program juga diperlukan intelektual matematika.

3.2 SARAN

Untuk memudahkan dalam membuat bentuk yang diinginkan, hal yang perlu diperkuat adalah kemampuan matematika supaya memudahkan dalam membuat bentuk yang diinginkan serta tak lupa pula ketepatan dalam menentukan titik koordinat.