**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное бюджетное образовательное учреждение высшего**

**профессионального образования**

**«Казанский национальный исследовательский технический университет**

**им. А.Н. Туполева-КАИ»**

Институт компьютерных технологий и защиты информации

Лабораторная работа № 2

по дисциплине «Компьютерная графика»

**Выполнил**

студент группы 4211

Абдуллаев Л.Х.

Казань

2020

# Задание 1

Цель : Произвести движение квадратов.

Листинг:

#include <Gl/glut.h>

#include <math.h>

GLfloat angle = 0;

GLfloat k = 1;

void init(void)

{

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(-15, 15, -15, 15);

//glClearColor(1.0, 1.0, 1.0, 0.0);

//glClear(GL\_COLOR\_BUFFER\_BIT);

}

void figure1(void)

{

glColor3f(0.0, 1.0, 0.0);

glBegin(GL\_QUADS); // движущийся зеленый квадрат

glVertex2f(1, 1);

glVertex2f(-1, 1);

glVertex2f(-1, -1);

glVertex2f(1, -1);

glEnd();

}

void figure2(void)

{

glColor3f(1.0, 0.0, 0.0);

glBegin(GL\_QUADS); //неподвижный красный квадрат

glVertex2f(6, 6);

glVertex2f(4, 6);

glVertex2f(4, 4);

glVertex2f(6, 4);

glEnd();

}

void axis(void) // рисование осей

{

glColor3f(0.0, 0.0, 0.0);

glBegin(GL\_LINES);

glVertex2f(0, -12);

glVertex2f(0, 12);

glVertex2f(-12, 0);

glVertex2f(12, 0);

glEnd();

}

void display(void) //функция рисования и обновления окна

{

glMatrixMode(GL\_MODELVIEW);//matrix

glLoadIdentity();//matrix1

glClearColor(1.0, 1.0, 1.0, 0.0);//color

//glClear(GL\_COLOR\_BUFFER\_BIT);//coloring the background

axis();

glTranslatef(5, 5, 0);

glRotatef(angle, 0, 0, 1);

glTranslatef(-5, -5, 0);

figure2();

glLoadIdentity();

glTranslatef(5, 5, 0);

glRotatef(-angle, 0, 0, 1);

glTranslatef(-5, -5, 0);

glScalef(k, k, 0);

figure1();

glFlush();

}

void timer(int value)

{

k -= 0.1;

if (k < 0.1) k = 1;

angle += 5;

if (angle >= 360)

{

k = 1;

angle = 0;

glClearColor(1.0, 1.0, 1.0, 0.0);

glClear(GL\_COLOR\_BUFFER\_BIT);

}

glutPostRedisplay();

glutTimerFunc(100, timer, 0);

}

void main(int argc, char \*\*argv)

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(500, 500);

glutInitWindowPosition(0, 0);

glutCreateWindow("Преобразования модели");

glutDisplayFunc(display);

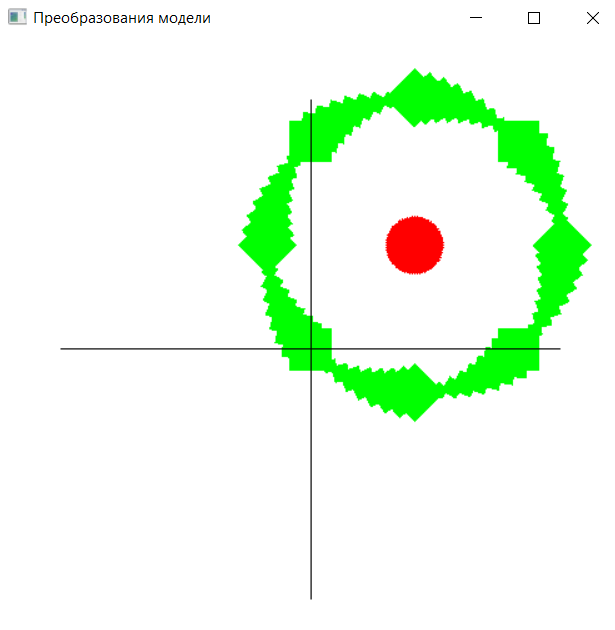
glutTimerFunc(500, timer, 0);

init();

glutMainLoop();

}

Результат работы:



# Задание 2

Цель : реализовать программу, работающую с масштабированием, вращением и перемещением объектов.

Листинг:

#include "pch.h"

#include <GL/glut.h>

#include <stdlib.h>

#include <math.h>

#include <windows.h>

#define PI 3.1459

GLfloat R = 640.0 / 480; //Форматное соотношение

GLfloat w = 40; //Ширина мирового окна

GLfloat h; //Высота мирового окна

GLfloat l, r, b, t; //Параметры мирового окна

GLfloat f = 30.0;

void init(void)

{ //Расчет параметров мирового окна

h = w / R; l = -w / 2; r = w / 2; b = -h / 2; t = h / 2;

glClearColor(1.0, 1.0, 0.0, 0.0);

glClear(GL\_COLOR\_BUFFER\_BIT);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(l, r, b, t);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void reshape(GLsizei W, GLsizei H)

{

if (R > W / H) glViewport(0, 0, W, W / R);

else glViewport(0, 0, H\*R, H);

}

void showAxis(void)

{

glColor3f(0.0f, 0.0f, 1.0f);

glBegin(GL\_LINES);

glVertex2f(0, 0);

glVertex2f(0, t);

glVertex2f(0, 0);

glVertex2f(r, 0);

glEnd();

}

void fig0(void)

{

glColor3f(0.0, 1.0, 0.0);

glBegin(GL\_LINE\_LOOP);

glVertex2f(0.0, 0.0);

glVertex2f(0.0, 3.0);

glVertex2f(5.0, 2.0);

glVertex2f(5.0, 0.0);

glEnd();

}

void scene(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT);

showAxis();

fig0();

glPushMatrix();

glTranslatef(4.5, 8.0, 0.0);

glRotatef(f, 0.0, 0.0, 1.0);

glScalef(1.0, -1.0, 1.0);

fig0();

glPopMatrix( );

glFlush();

glutSwapBuffers();

f+=30; if(f==360) f=0;

Sleep(200);

}

void main(int argc, char \*\*argv)

{

glutInit(&argc, argv);

//glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowSize(640, 480);

glutInitWindowPosition(20, 20);

glutCreateWindow("Myprog");

glutReshapeFunc(reshape);

glutDisplayFunc(scene);

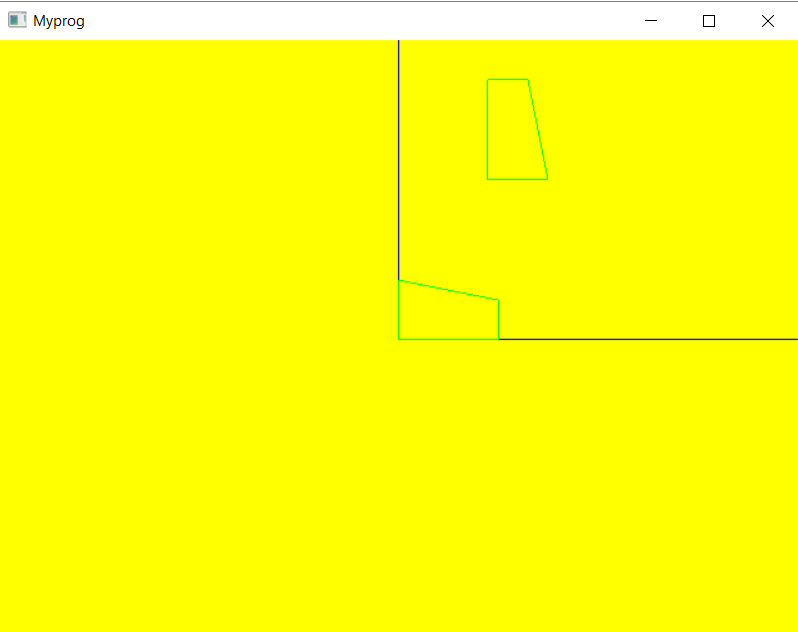
glutIdleFunc(scene);

init();

glutMainLoop();

}

Результат работы:



# Задание 3

Цель : создать программу, которая рисует сцену, состоящую из квадрата и нескольких треугольников.

Листинг:

#include "pch.h"

#include <GL/glut.h>

#include <stdlib.h>

#include <math.h>

#include <windows.h>

void init(void)

{

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(-15, 15, -15, 15);

glClearColor(1.0, 1.0, 1.0, 0.0);

glClear(GL\_COLOR\_BUFFER\_BIT);

}

void figure1(void) // треугольники

{

glColor3f(0.0, 0.0, 1.0);

glBegin(GL\_TRIANGLES);

glVertex2f(0, 5);

glVertex2f(-2.5, 0);

glVertex2f(2.5, 0);

glEnd();

}

void figure2(void) // квадрат

{

glColor3f(1.0, 1.0, 0.0);

glBegin(GL\_QUADS);

glVertex2f(5, 5);

glVertex2f(-5, 5);

glVertex2f(-5, -5);

glVertex2f(5, -5);

glEnd();

}

void axis2(void) // оси треугольников

{

glColor3f(0.0, 0.0, 0.0);

glBegin(GL\_LINES);

glVertex2f(0, 0);

glVertex2f(0, 6);

glVertex2f(-3.5, 0);

glVertex2f(0, 0);

glEnd();

}

void axis(void) // оси квадрата

{

glColor3f(0.0, 0.0, 0.0);

glBegin(GL\_LINES);

glVertex2f(0, -12);

glVertex2f(0, 12);

glVertex2f(-12, 0);

glVertex2f(12, 0);

glEnd();

}

void display(void) //функция рисования и обновления окна

{

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glClear(GL\_COLOR\_BUFFER\_BIT);

figure2();

axis();

glTranslatef(0, 5, 0);

figure1();

axis2();

glLoadIdentity();

glTranslatef(-5, 0, 0);

glRotatef(90, 0, 0, 1);

figure1();

axis2();

glLoadIdentity();

glTranslatef(0, -5, 0);

glRotatef(180, 0, 0, 1);

figure1();

axis2();

glLoadIdentity();

glTranslatef(5, 0, 0);

glRotatef(270, 0, 0, 1);

axis2();

figure1();

glFlush();

}

int main(int argc, char \*\*argv)

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(500, 500);

glutInitWindowPosition(0, 0);

glutCreateWindow("n3");

glutDisplayFunc(display);

init();

glutMainLoop();

}

Результат работы:

