Лабораторная работа №2

4210\_БеловВА  
Листинг 1:  
#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;

int count = 1;

void init(void)

{

h = w / R; l = -w / 2; r = w / 2; b = -h / 2; t = h / 2; //Расчет параметров мирового окна

glClearColor(1.0, 1.0, 1.0, 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 fig0(void)

{

double a = 0.0, b = 5.0, c = 7.0;

if (count % 2 == 0) {

b += 0.5;

c += 0.5;

}

else {

b -= 0.5;

c -= 0.5;

}

count++;

glColor3f(0.0, 1.0, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(a, a);

glVertex2f(a, b);

glVertex2f(c, b);

glVertex2f(c, a);

glEnd();

}

void scene(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glPushMatrix();

glTranslatef(4.5, 3.0, 0.0);

glRotatef(f, 0.0, 0.0, 1.0);

glScalef(1.0, -1.0, 1.0);

fig0();

glPopMatrix();

glFlush();

glutSwapBuffers();

f += 5; if (f == 360) f = 0;

Sleep(70);

}

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);

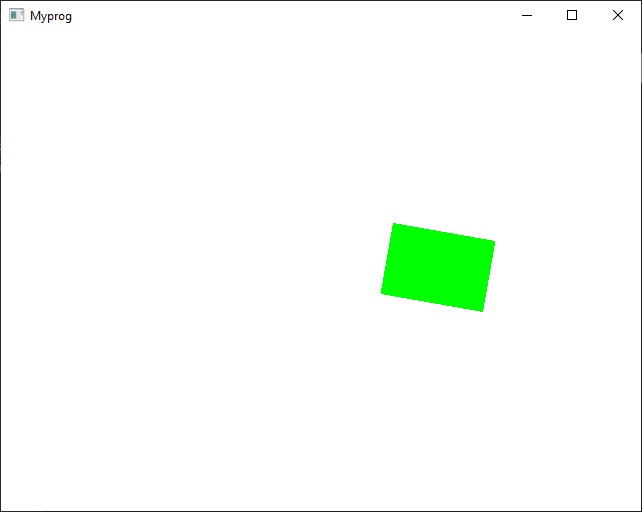
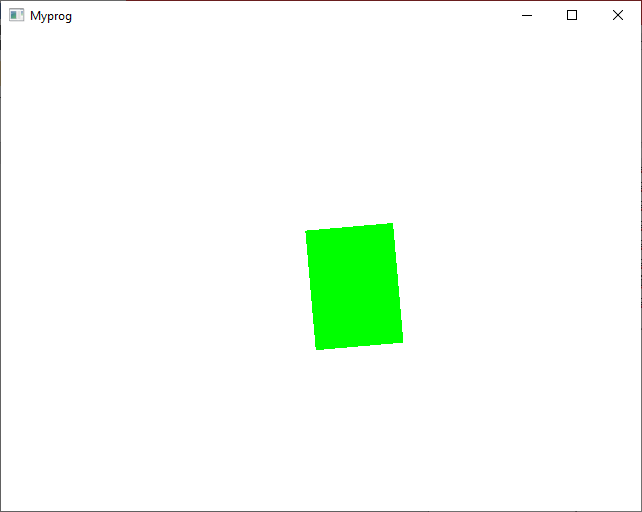
glutIdleFunc(scene);

init();

glutMainLoop();

}

Результат программы:

  
  
  
  
  
  
  
Задание2:  
#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 = 0;

int count = 1;

void init(void)

{

h = w / R; l = -w / 2; r = w / 2; b = -h / 2; t = h / 2; //Расчет параметров мирового окна

glClearColor(1.0, 1.0, 1.0, 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 (H != 0) {

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

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

}

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)

{

double a = -7.0, b = 7.0, c = 7.0;

glColor3f(0.0, 1.0, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(a, a);

glVertex2f(a, b);

glVertex2f(c, b);

glVertex2f(c, a);

glEnd();

}

void fig1(void)

{

double a = 0.0, b = 3.0, c = 6.0;

glColor3f(0.0, 1.0, 0.0);

glBegin(GL\_POLYGON);

glVertex2f(a, a);

glVertex2f(a, b);

glVertex2f(c, b);

glVertex2f(c, a);

glEnd();

}

void scene(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT);

fig0();

if (f == 0) {

glPushMatrix();

glTranslatef(7, -7, 0.0);

glRotatef(-60, 0.0, 0.0, 1.0);

glScalef(1.0, 1.0, 1.0);

fig1();

}

showAxis();

if (f == 1) {

glPushMatrix();

glTranslatef(7, 7, 0.0);

glRotatef(60, 0.0, 0.0, 1.0);

glScalef(1.0, 1.0, 1.0);

fig1();

showAxis();

glPopMatrix();

}

showAxis();

glPopMatrix( );

if (f == 2) {

glPushMatrix();

glTranslatef(-7, 7, 0.0);

glRotatef(-45, 0.0, 0.0, 1.0);

glScalef(-1.0, 1.0, 1.0);

fig1();

showAxis();

glPopMatrix();

}

showAxis();

if (f == 3) {

glPushMatrix();

glTranslatef(-7, -7, 0.0);

glRotatef(30, 0.0, 0.0, 1.0);

glScalef(-1.0, 1.0, 1.0);

fig1();

showAxis();

glPopMatrix();

}

showAxis();

glFlush();

glutSwapBuffers();

f ++; if (f == 4) f = 0;

Sleep(500);

}

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 программы:  
