

**КАЗАНСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМ. А. Н. ТУПОЛЕВА – КАИ**

Институт компьютерных технологий и защиты информации

Кафедра: Автоматизированных систем обработки информации и управления

Г.М. НАБЕРЕЖНОВ, Н.Н. МАКСИМОВ

ЭЛЕМЕНТАРНОЕ ПРОГРАММИРОВАНИЕ ГРАФИКИ В OpenGL

**Методическое пособие
к лабораторным работам по курсу
«Компьютерная геометрия и графика»**

Студент, гр. 4210

Гауиш М.Г

Преподаватель

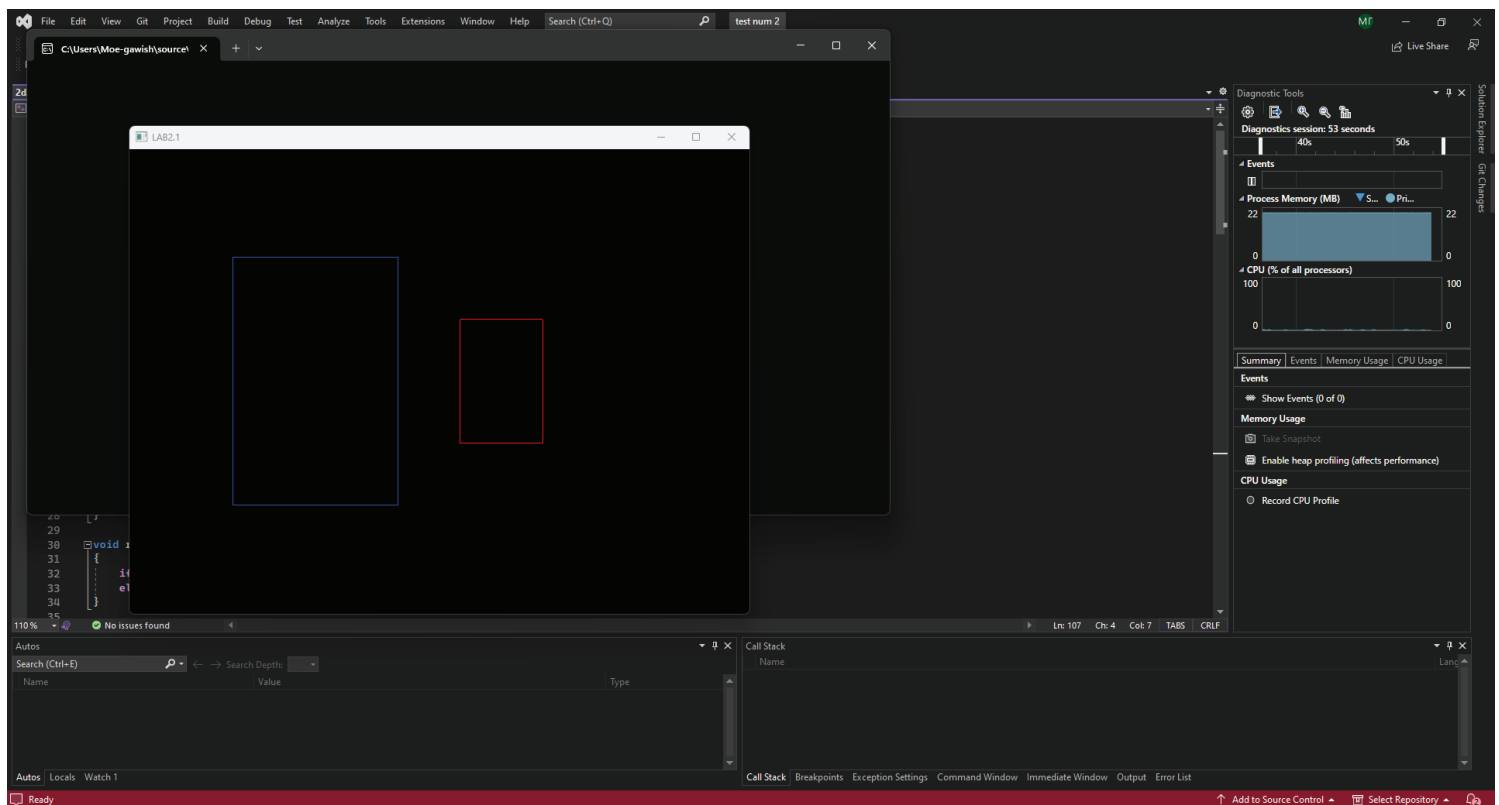
Гаптуллазянова Гульшат Ильдусовна

Казань – 2023

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО ВЫПОЛНЕНИЯ

1. Используя операции преобразования модели, создайте программу:
- 1) Прямоугольник вращается вокруг произвольно заданной точки и пульсирует;
 - 2) Прямоугольник, произвольно заданный в МСК, масштабируется относительно своего центра и пульсирует;
 - 3) Прямоугольник, произвольно заданный в МСК, зеркально отображается относительно произвольно заданной прямой и пульсирует;
 - 4) Точка вращается вокруг начала координат по сходящейся спирали;
 - 5) Точка вращается вокруг произвольной точки по сходящейся спирали;
 - 6) Две произвольные окружности пульсируют в противофазе.
- Все размерные величины подберите так, чтобы на экране хорошо просматривалось изображение и его особенности.

Скриншот работы программы:



```

1  #include <iostream>
2  #include <stdio.h>
3  #define GL_SILENCE_DEPRECATION
4  #include <GL/glut.h>
5  #include <cmath>
6  #include <stdlib.h>
7
8  #define PI 3.1459
9  GLfloat R = 800.0 / 600; //Форматное соотношение
10 GLfloat w = 600; //Ширина мирового окна
11 GLfloat h; //Высота мирового окна
12 GLfloat l, r, b, t; //Параметры мирового окна
13 GLfloat f = 30.0;
14 GLfloat x = 0;
15 GLfloat x1 = 2;
16 int mode = 1, model = 2;
17
18 void init(void)
19 {
20     h = w / R; l = -w / 2; r = w / 2; b = -h / 2; t = h / 2; //Расчет параметров миро-вого окна
21     glClearColor(0.0, 0.0, 0.0, 0.0);
22     glClear(GL_COLOR_BUFFER_BIT);
23     glMatrixMode(GL_PROJECTION);
24     glLoadIdentity();
25     gluOrtho2D(l, r, b, t);
26     glMatrixMode(GL_MODELVIEW);
27     glLoadIdentity();
28 }
29
30 void reshape(GLsizei W, GLsizei H)
31 {
32     if (R > W / H) glViewport(0, 0, W, W / R);
33     else glViewport(0, 0, H * R, H);
34 }
35
36 void fig0(void)
37 {
38     glColor3f(0.0, 0.0, 1.0);
39     glBegin(GL_LINE_LOOP);
40     glVertex2f(-10.0, -6.0);
41     glVertex2f(-10.0, 6.0);
42     glVertex2f(-2.0, 6.0);
43     glVertex2f(-2.0, -6.0);

```

```

44     glEnd();
45 }
46 void fig1(void)
47 {
48     glColor3f(1.0, 0.0, 0.0);
49     glBegin(GL_LINE_LOOP);
50     glVertex2f(10.0, 6.0);
51     glVertex2f(10.0, -6.0);
52     glVertex2f(2.0, -6.0);
53     glVertex2f(2.0, 6.0);
54     glEnd();
55 }
56
57 void scene(void)
58 {
59     glClear(GL_COLOR_BUFFER_BIT);
60     glPushMatrix();
61     glScalef(x, -x, x);
62     fig0();
63     glPopMatrix();
64     glFlush();
65     glutSwapBuffers();
66     //x+=0.5; if(x==2.0) x=0;
67     if (mode == 1)
68     {
69         x += 0.5;
70     }
71     else if (mode == 2)
72     {
73         x -= 0.5;
74     }
75     if (x == 2.0)
76     {
77         mode = 2;
78     }
79     else if (x == 0.5)
80     {
81         mode = 1;
82     }
83     glPushMatrix();
84     glScalef(x1, -x1, x1);
85     fig1();
86

```

10% No issues found Ln: 107 Ch: 4 Col: 7 TABS CRLF

10% No issues found Ln: 107 Ch: 4 Col: 7 TABS CRLF

```

87     glPopMatrix();
88     glFlush();
89     glutSwapBuffers();
90     //x1-=0.5; if(x1==0.0) x1=2.0;
91     if (model == 1)
92     {
93         x1 += 0.5;
94     }
95     else if (model == 2)
96     {
97         x1 -= 0.5;
98     }
99     if (x1 == 2.0)
100    {
101        model = 2;
102    }
103    else if (x1 == 0.5)
104    {
105        model = 1;
106    }
107    //sleep(100000);
108 }
109
110 int main(int argc, char** argv)
111 {
112     glutInit(&argc, argv);
113     glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
114     glutInitWindowSize(800, 600);
115     glutInitWindowPosition(150, 150);
116     glutCreateWindow("LAB2.1");
117     glutReshapeFunc(reshape);
118     glutDisplayFunc(scene);
119     glutIdleFunc(scene);
120     init();
121     glutMainLoop();
122 }

```

Листинг:

```

#include <iostream>
#include <stdio.h>
#define GL_SILENCE_DEPRECATION
#include <GL/glut.h>
#include <cmath>
#include <stdlib.h>

#define PI 3.1459
GLfloat R = 600 / 800.0; //Форматное соотношение
GLfloat w = 60; //Ширина мирового окна
GLfloat h; //Высота мирового окна
GLfloat l, r, b, t; //Параметры мирового окна
GLfloat f = 30.0;
GLfloat x = 0;
GLfloat x2 = 1;
int mode = 1, mode2 = 1;

void init(void)
{
    h = w / R; l = -w / 2; r = w / 2; b = -h / 2; t = h / 2; //Расчет параметров миро-вого окна
    glClearColor(0.0,0.0,0.0,0.0);
    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 fig0(void)
{
    glColor3f(1.0 ,0.0 ,0.0);
    glBegin(GL_LINE_LOOP);
    glVertex2f(-6.0- ,10.0);
    glVertex2f(-6.0 ,10.0);
    glVertex2f(-6.0 ,2.0);
    glVertex2f(-6.0- ,2.0);
    glEnd();
}
```

```
void fig1(void)
{
    glColor3f(0.0 ,0.0 ,1.0);
    glBegin(GL_LINE_LOOP);
    glVertex2f(6.0 ,10.0);
    glVertex2f(6.0- ,10.0);
    glVertex2f(6.0- ,2.0);
    glVertex2f(6.0 ,2.0);
    glEnd();
}
```

```
void scene(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glPushMatrix();
    glScalef(x, -x, x);
    fig0();
    glPopMatrix();
    glFlush();
    glutSwapBuffers();
    //x+=0.5; if(x==2.0) x=0;
    if (mode == 1)
    {
        x += 0.5;
    }
    else if (mode == 2)
    {
        x -= 0.5;
    }
    if (x == 2.0)
    {
        mode = 2;
    }
}
```

```

else if (x == 0.5)
{
    mode = 1;
}

glPushMatrix();
glScalef(x1, -x1, x1);
fig1();
glPopMatrix();
glFlush();
glutSwapBuffers();
//x0.5=-1; if(x0.0==1) x2.0=1;
if (mode1 == 1)
{
    x0.5 = + 1;
}
else if (mode2 == 1)
{
    x0.5 = - 1;
}
if (x2.0 == 1)
{
    mode2 = 1;
}
else if (x0.5 == 1)
{
    mode1 = 1;
}
//usleep(100000);
}

```

```

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(600 ,800);
    glutInitWindowPosition(150 ,150);
    glutCreateWindow("LAB 2 First task");
    glutReshapeFunc(reshape);
    glutDisplayFunc(scene);
    glutIdleFunc(scene);
    init();
    glutMainLoop();
}

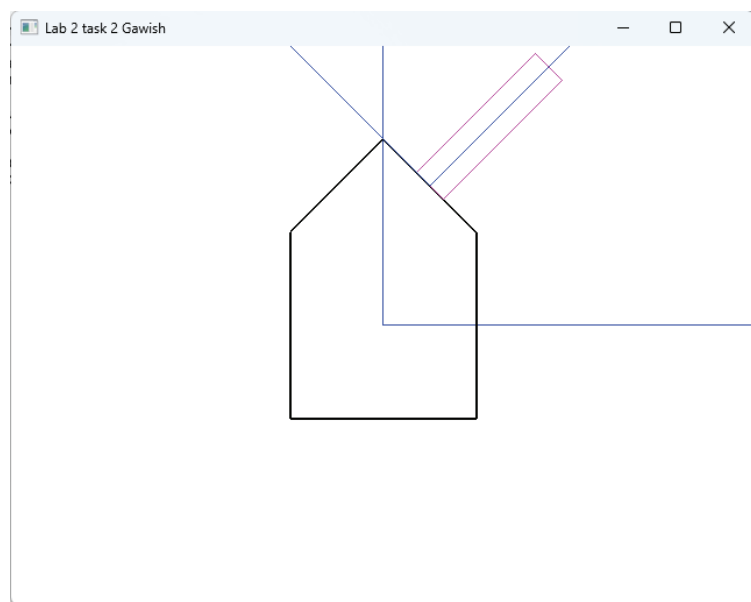
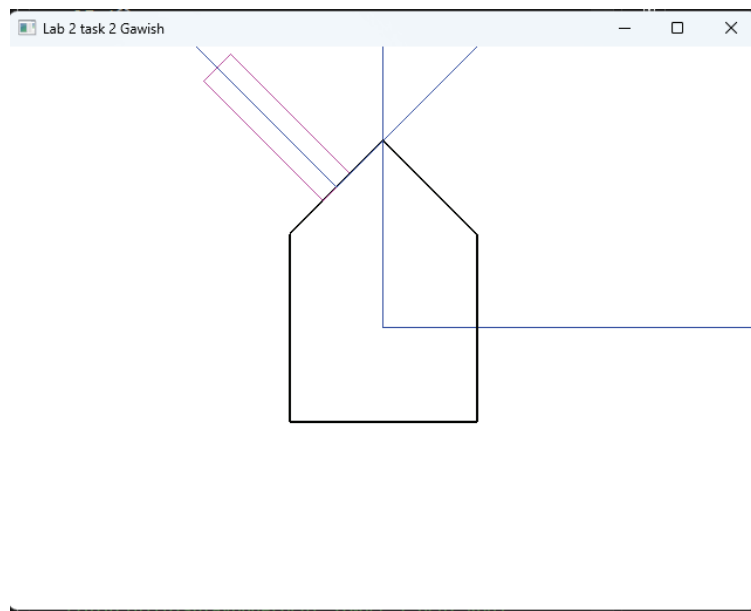
```

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО ВЫПОЛНЕНИЯ

2-Для вариантов, приведенных ниже, создайте программу, которая рисует сцену, состоящую из фиг.0 и нескольких фиг.1. Для фиг.1 создайте отдельную функцию, описывающую ее в СКО, и затем перемещайте фиг.1 из позиции в позицию. В отчете для каждой позиции фиг.1 выпишите последовательность движений фиг.1, в результате которых она попадает в заданную позицию. Кроме того, считайте, что функция для фиг.1 описывает ее в локальной системе координат СКЛ, а полученные преобразования понимайте как преобразования координат из СКЛ в СКО. Для каждой позиции фиг.1. изобразите цепочку преобразований систем координат и все координатные фреймы как это сделано на рис.2.

Примечание. Fig.0 и Fig.1 во всех позициях должны быть нарисованы с положительными координатными полуосьми (стрелки не рисовать).

Скриншот работы программы 5 вар:



```

1  #include <GL/glut.h>
2  #include <stdlib.h>
3  #include <math.h>
4  #include <windows.h>
5
6
7  #define PI 3.1459
8
9  GLfloat R = 640.0 / 480; //Форматное соотношение
10 GLfloat w = 480; //Ширина мирового окна
11 GLfloat h; //Высота мирового окна
12 GLfloat l, r, b, t; //Параметры мирового окна
13 GLfloat f = 45.0;
14 GLfloat tr_x = 2.5;
15 GLfloat sc_x = -1.0;
16 GLfloat rt_a = -45;
17
18 void init(void)
19 {
20     h = w / R; l = -w / 2; r = w / 2; b = -h / 2; t = h / 2; //Расчет параметров мирового окна
21     glClearColor(1.0, 1.0, 1.0, 0.0);
22     glClear(GL_COLOR_BUFFER_BIT);
23     glMatrixMode(GL_PROJECTION);
24     glLoadIdentity();
25     gluOrtho2D(l, r, b, t);
26     glMatrixMode(GL_MODELVIEW);
27     glLoadIdentity();
28 }
29
30 void reshape(GLsizei W, GLsizei H)
31 {
32     if (R > W / H) glViewport(0, 0, W, W / R);
33     else glViewport(0, 0, H * R, H);
34 }
35
36 void showAxis(void)
37 {
38     glColor3f(0.0f, 0.0f, 1.0f);
39     glBegin(GL_LINES);
40     glVertex2f(0, 0);
41     glVertex2f(0, t);
42     glVertex2f(0, 0);
43     glVertex2f(r, 0);
44     glEnd();
45 }
46
47
48 void fig0(void)
49 {
50     glColor3f(0.0f, 0.0f, 0.0f);
51     glLineWidth(2);
52     glBegin(GL_LINE_LOOP);
53     glVertex2d(5.0, -5.0);
54     glVertex2d(5.0, 5.0);
55     glVertex2d(0.0, 10.0);
56     glVertex2d(-5.0, 5.0);
57     glVertex2d(-5.0, -5.0);
58     glEnd();
59 }
60
61 void fig1(void)
62 {
63     glColor3f(1.0, 0.0, 1.0);
64     glLineWidth(1);
65     glBegin(GL_LINE_LOOP);
66     glVertex2d(-1.0, 0.0);
67     glVertex2d(1.0, 0.0);
68     glVertex2d(1.0, 3.0);
69     glVertex2d(-1.0, 3.0);
70     glEnd();
71     glColor3f(0.0f, 0.0f, 1.0f);
72     glBegin(GL_LINES);
73     glVertex2f(0, 0);
74     glVertex2f(0, t);
75     glVertex2f(0, 0);
76     glVertex2f(r, 0);
77     glEnd();
78 }
79
80
81
82 void scene(void)
83 {
84     glClear(GL_COLOR_BUFFER_BIT);
85     showAxis();
86     fig0();

```



```

87     glPushMatrix();
88     glTranslatef(tr_x, 7.5, 0.0);
89     glRotatef(rt_a, 0.0, 0.0, 1.0);
90     glScalef(sc_x, 3.0, 1.0);
91     fig1();
92     glPopMatrix();
93     glFlush();
94     glutSwapBuffers();
95     tr_x = -tr_x;
96     sc_x = -sc_x;
97     rt_a = -rt_a;
98     Sleep(500);
99 }
100
101 void main(int argc, char** argv)
102 {
103     glutInit(&argc, argv);
104     //glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
105     glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
106     glutInitWindowSize(640, 480);
107     glutInitWindowPosition(20, 20);
108     glutCreateWindow("Lab 2 task 2 Gawish");
109     glutReshapeFunc(reshape);
110     glutDisplayFunc(scene);
111     glutIdleFunc(scene);
112     init();
113     glutMainLoop();
114 }
115

```

Листинг:

```

#include <GL/glut.h>
#include <stdlib.h>
#include <math.h>
#include <windows.h>

#define PI 3.1459

GLfloat R = 480 / 640.0; //Форматное соотношение
GLfloat w = 40; //Ширина мирового окна
GLfloat h; //Высота мирового окна
GLfloat l, r, b, t; //Параметры мирового окна
GLfloat f = 45.0;
GLfloat tr_x = 2.5;
GLfloat sc_x = -1.0;
GLfloat rt_a = -45;

void init(void)
{
    h = w / R; l = -w / 2; r = w / 2; b = -h / 2; t = h / 2; //Расчет параметров мирового окна
    glClearColor(0.0, 1.0, 1.0, 1.0);
    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(0.0f, 0.0f, 0.0f);
    glLineWidth(2);
    glBegin(GL_LINE_LOOP);
    glVertex2d(5.0, 5.0);
    glVertex2d(5.0, 5.0);
    glVertex2d(10.0, 0.0);
    glVertex2d(-5.0, 5.0);
    glVertex2d(-5.0, 5.0);
    glEnd();
}

void fig1(void)
{
    glColor3f(1.0, 0.0, 1.0);
    glLineWidth(1);
    glBegin(GL_LINE_LOOP);
    glVertex2d(-0.0, 1.0);
    glVertex2d(0.0, 1.0);
    glVertex2d(3.0, 1.0);
    glVertex2d(-3.0, 1.0);
    glEnd();
    glColor3f(0.0f, 0.0f, 1.0f);
    glBegin(GL_LINES);
    glVertex2f(0, 0);
    glVertex2f(0, t);
}

```

```

    glVertex2f(0,0);
    glVertex2f(r, 0);
    glEnd();
}

void scene(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    showAxis();
    fig0();
    glPushMatrix();
    glTranslatef(tr_x, 0.0 ,7.5);
    glRotatef(rt_a, 1.0 ,0.0 ,0.0);
    glScalef(sc_x, 1.0 ,3.0);
    fig1();
    glPopMatrix();
    glFlush();
    glutSwapBuffers();
    tr_x = -tr_x;
    sc_x = -sc_x;
    rt_a = -rt_a;
    Sleep(500);
}

```

```

void main(int argc, char** argv)
{
    glutInit(&argc, argv);
    //glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
    glutInitWindowSize(480 ,640);
    glutInitWindowPosition(20 ,20);
    glutCreateWindow("Lab 2 task 2 Gawish");
    glutReshapeFunc(reshape);
    glutDisplayFunc(scene);
    glutIdleFunc(scene);
    init();
    glutMainLoop();
}

```

Библиотека Glut и opengl использовалась.

Большое спасибо)