

## Лабораторная работа №1 «Моделирование 2D объектов».

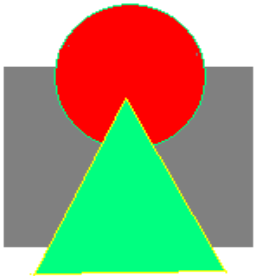
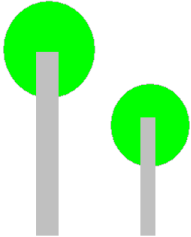
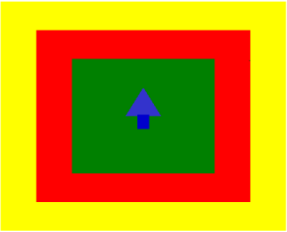

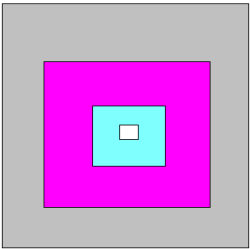
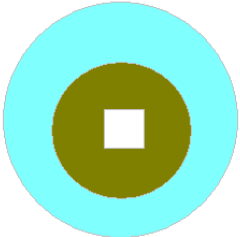
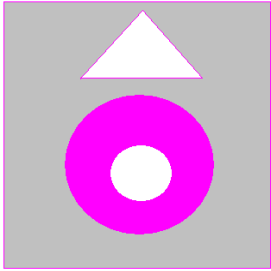
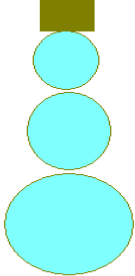


1. Построить прямоугольник. Листинг программы приведен в приложении 1. Изменить параметры освещенности, цвета объекта, фона и размера в соответствии с номером варианта (таб.1).
2. В соответствии со своим вариантом (табл. 2) построить 2D объекты.
3. Составить и защитить отчет по работе.

*Таблица 1. Изменение параметров к лабораторной работе № 1*

№ вар.	Задание	№ вар	Задание
1	положение - 3,3,3,0.9 направление света - -30,-1,-1 объект -1.0, 1.0, 0.1 фон - 3399CC 10x5	6	положение - 0,0,2,1 направление света - -2,-20,-2 объект - FF6666 фон – 1.0,1.0,0.5 5x9
2	положение - 1,1,10,0.8 направление света - 3,1,1 объект - FFCC66 фон - 330066 4x2	7	положение - 4,4,5,1 направление света - -3,-3,-3 объект - 3333CC фон – 0,0,0 7x3
3	положение - 3,3,5,1 направление света - -30,-1,-1 объект - 3399CC фон - CC3399 3x8	8	положение - 3,0,3,0.1 направление света - 5,5,4 объект -CCCCFF фон - CC0033 8x4
4	положение - 2,2,4,0.9 направление света - -30,-1,-1 объект -FFFFCC фон - 102.0.51 1x7	9	положение - 0,10,3,0.8 направление света - -4,-4,10 объект -255.204.153 фон - 153.51.0 7x2
5	положение - 3,5,5,0.2 направление света - -30,-1,-1 объект -255.153.204 фон - 102.51.153 10x8	10	положение - 4,8,2,1 направление света - -4,-4,-4 объект -CCFFFF фон - 999999 2x6

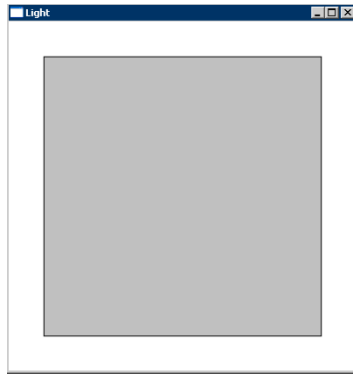
*Таблица 2. Исходные задания к лабораторной работе № 1*

№ вар	Задание	№ вар	Задание
-------	---------	-------	---------

1		6	
2		7	
3		8	
4		9	
5		10	

Приложение 1. Программа «Построение квадрата»

**Результат выполнения программы**



## Текст программы

```
#include <windows.h>

#include <GL/gl.h>

#include <GL/glu.h>

#include <GL/glaux.h>

void CALLBACK resize(int width,int height)

{

    glViewport(0,0,width,height);

    glMatrixMode( GL_PROJECTION );

    glLoadIdentity();

    glOrtho(-5,5, -5,5, 2,12);

    gluLookAt( 0,0,5, 0,0,0, 0,1,0 );

    glMatrixMode( GL_MODELVIEW );

}

void CALLBACK display(void)

{

    glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    glColor3d(1,1,1);

    glBegin(GL_QUADS);

        glVertex3d(-4,-4,0);

        glVertex3d(-4, 4,0);

        glVertex3d( 4, 4,0);
```

```

    glVertex3d( 4,-4,0);

glEnd();

auxSwapBuffers();

}

void main()

{

float pos[4] = { 3,3,3,0.5};

float dir[3] = { -1,-1,-1 };

    GLfloat mat_specular[] = { 1,1,1,1 };

    auxInitPosition( 50, 10, 400, 400);

    auxInitDisplayMode( AUX_RGB | AUX_DEPTH | AUX_DOUBLE );

    auxInitWindow( "Light" );

    auxIdleFunc(display);

    auxReshapeFunc(resize);

    glEnable(GL_DEPTH_TEST);

    glEnable(GL_COLOR_MATERIAL);

    glEnable(GL_LIGHTING);

    glEnable(GL_LIGHT0);

    glLightfv(GL_LIGHT0, GL_POSITION, pos);

    glLightfv(GL_LIGHT0, GL_SPOT_DIRECTION, dir);

    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);

    glMaterialf(GL_FRONT, GL_SHININESS, 128.0);

    glLighti(GL_LIGHT0, GL_SPOT_EXPONENT, 0);

    glLighti(GL_LIGHT0, GL_SPOT_CUTOFF, 90);

auxMainLoop(display);

}

```