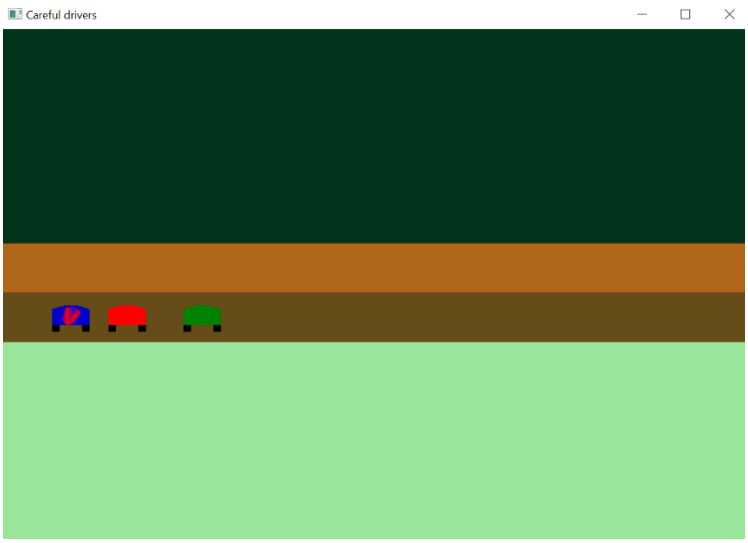
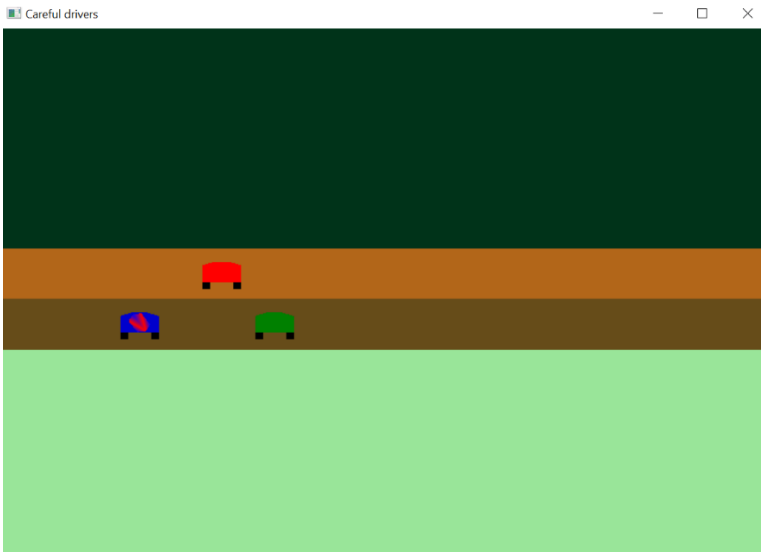
Stratan Ruxandra-Ioana
343

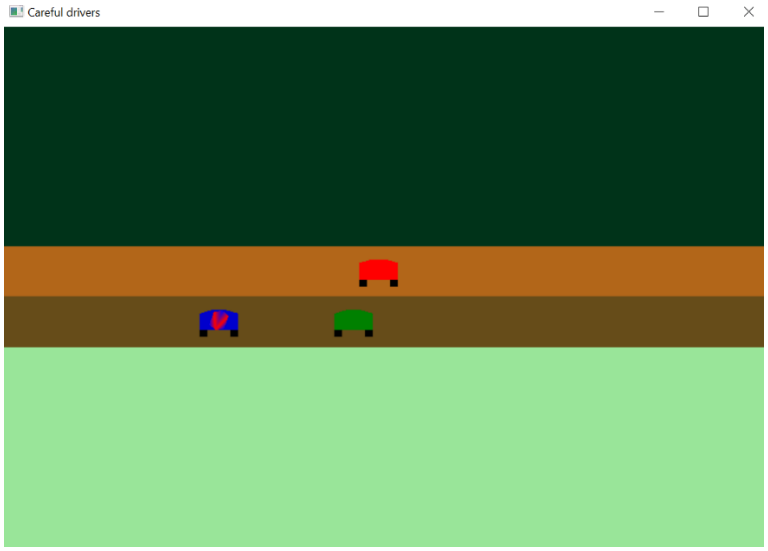
Proiect 1

Tema:

- Depasire
- Scena prezinta trei masini care se deplaseaza pe un drum drept ; ultima este o masina de politie cu girofar care urmareste a doua masina, iar aceasta o va depasi pe prima







Transformari :

- Translatii

```
resizeMatrix = glm::ortho(-width, width, -height, height);
matrTrans1 = glm::translate(glm::mat4(1.0f), glm::vec3(i + 40.0f, j + 80.0f, 0.0));
matrTrans2 = glm::translate(glm::mat4(1.0f), glm::vec3(ii + 40.0f, jj + 80.0f, 0.0));
```

- matrTrans1 controleaza miscarea celei de-a doua masini, care isi schimba directia in momentul in care atinge unele coordonate, detaliate in cadrul functiei de miscare move(); de asemenea, masina este mai rapida pe anumite portiuni

```
void move() {
    if (i < -25.0f) {
        i += 0.02f;
        ii += 0.02f;
        angle += 0.01;
    }
    else if (i < -10.0f) {
        i += 0.03f;
        j += 0.03f;
        ii += 0.02f;
        angle += 0.01;
    }
    else if (i < 45.0f) {
        i += 0.04f;
        ii += 0.02f;
        angle += 0.01;
    }
    else if (i < 60.0f) {
        i += 0.03f;
        j -= 0.03f;
        ii += 0.02f;
        angle += 0.01;
    }
    else if (i < 100.0f) {
        i += 0.05f;
        ii += 0.02f;
        angle += 0.01;
```

No issues found

- matrTrans2 controleaza miscarea celorlalte doua masini, care nu isi schimba directia si au viteza constanta
- rotile preiau matricea de translatie a masinii careia apartin

- Rotatie :

```
matrTrans2 = glm::translate(glm::mat4(1.0f), glm::vec3(ii + 40.0f, jj + 80.0f,
matrRot = glm::rotate(glm::mat4(1.0f), angle, glm::vec3(0.0, 0.0, 1.0));
matrDepl1 = glm::translate(glm::mat4(1.0f), glm::vec3(90.0f, 17.0f, 0.0f));
matrDepl2 = glm::translate(glm::mat4(1.0f), glm::vec3(-90.0f, -17.0f, 0.0f));
```

```
//Girofar
myMatrix = resizeMatrix * matrTrans2 * matrDepl2 * matrRot * matrDepl1;
codCol = 0;
glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
glUniform1i(codColLocation, codCol);
glDrawArrays(GL_POLYGON, 42, 6);
glutSwapBuffers();
```

- Se aplica rotatia si translatia pe girofarul situat pe prima masina

Elemente de originalitate – design

- Fiecare masina are forma de hexagon, la care se adauga cate doua roti in forma de patrat care urmeaza aceeasi traiectorie

- ‘Girofarul’ primei masini este tot un hexagon cu varfurile colorate alternativ rosu si albastru
- Fundalul este reprezentat de doua benzi de drum care au nuante diferite de maro si de doua campuri cu nuante diferite de verde in partea de sus, respectiv jos, a ferestrei
- Scena reprezinta o depasire in cadrul unei urmariri

Resurse:

- Codurile de la laborator (03_02_animatie_new.cpp ; 03_05_transformari_compunere.cpp)

Anexe:

1) Codul sursa (Proiect01.cpp)

```
#include <windows.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <iostream>
#include <GL/glew.h>
#include <GL/freeglut.h>
#include "loadShaders.h"

#include "glm/glm.hpp"
#include "glm/gtc/matrix_transform.hpp"
#include "glm/gtx/transform.hpp"
#include "glm/gtc/type_ptr.hpp"

using namespace std;

GLuint
	VaoId,
	VboId,
	ColorBufferId,
	ProgramId,
	myMatrixLocation,
	matrScaleLocation,
	matrTranslLocation,
```

```
	matrRotlLocation,
	codColLocation;

int codCol;
float PI = 3.141592, angle = 0.0;
float tx = 0; float ty = 0;
float width = 100, height = 100;
float i, j,ii,jj;
glm::mat4
	myMatrix, resizeMatrix, matrTransl, matrTrans2, matrRot, matrDepl1, matrDepl2;

void displayMatrix()
{
	for (int ii = 0; ii < 4; ii++)
	{
		for (int jj = 0; jj < 4; jj++)
			cout << myMatrix[ii][jj] << "  ";
		cout << endl;
	};
	cout << "\n";
};



void move() {
	if (i < -25.0f) {
		i += 0.02f;
		ii += 0.02f;
		angle += 0.01;
	}
	else if (i < -10.0f) {
		i += 0.03f;
		j += 0.03f;
		ii += 0.02f;
		angle += 0.01;
	}
	else if (i < 45.0f) {
		i += 0.04f;
		ii += 0.02f;
```

```
		angle += 0.01;
	}
	else if (i < 60.0f) {
		i += 0.03f;
		j -= 0.03f;
		ii += 0.02f;
		angle += 0.01;
	}
	else if (i < 100.0f) {
		i += 0.05f;
		ii += 0.02f;
		angle += 0.01;
	}
	else {
		i = -40.0f;
		j = -50.0f;
		ii = -40.0f;
		angle += 0.01;
	}
	glutPostRedisplay();
}


void CreateVBO(void)
{
	GLfloat Vertices[] = {
	//masina albastra
	-95.0f,  -20.0f, 0.0f, 1.0f,
	-85.0f, -20.0f, 0.0f, 1.0f,
	-85.0f,  -15.0f, 0.0f, 1.0f,
	-88.0f, -14.0f, 0.0f, 1.0f,
	-92.0f, -14.0f, 0.0f, 1.0f,
	-95.0f,  -15.0f, 0.0f, 1.0f,
	//roata A1
	-95.0f,  -20.0f, 0.0f, 1.0f,
	-93.0f,  -20.0f, 0.0f, 1.0f,
	-93.0f,  -22.0f, 0.0f, 1.0f,
	-95.0f,  -22.0f, 0.0f, 1.0f,
	//roata A2
```

```
-87.0f,  -20.0f, 0.0f, 1.0f,
-85.0f,  -20.0f, 0.0f, 1.0f,
-85.0f,  -22.0f, 0.0f, 1.0f,
-87.0f,  -22.0f, 0.0f, 1.0f,
// masina rosie
-80.0f,  -20.0f, 0.0f, 1.0f,
-70.0f, -20.0f, 0.0f, 1.0f,
-70.0f,  -15.0f, 0.0f, 1.0f,
-73.0f, -14.0f, 0.0f, 1.0f,
-77.0f, -14.0f, 0.0f, 1.0f,
-80.0f,  -15.0f, 0.0f, 1.0f,
//roata R1:
-80.0f,  -20.0f, 0.0f, 1.0f,
-78.0f,  -20.0f, 0.0f, 1.0f,
-78.0f,  -22.0f, 0.0f, 1.0f,
-80.0f,  -22.0f, 0.0f, 1.0f,
//roata R2
-72.0f,  -20.0f, 0.0f, 1.0f,
-70.0f,  -20.0f, 0.0f, 1.0f,
-70.0f,  -22.0f, 0.0f, 1.0f,
-72.0f,  -22.0f, 0.0f, 1.0f,
//masina verde
-60.0f,  -20.0f, 0.0f, 1.0f,
-50.0f, -20.0f, 0.0f, 1.0f,
-50.0f,  -15.0f, 0.0f, 1.0f,
-53.0f,  -14.0f, 0.0f, 1.0f,
-57.0f,  -14.0f, 0.0f, 1.0f,
-60.0f,  -15.0f, 0.0f, 1.0f,
//roata V1
-60.0f,  -20.0f, 0.0f, 1.0f,
-58.0f,  -20.0f, 0.0f, 1.0f,
-58.0f,  -22.0f, 0.0f, 1.0f,
-60.0f,  -22.0f, 0.0f, 1.0f,
//roata V2
-52.0f,  -20.0f, 0.0f, 1.0f,
-50.0f,  -20.0f, 0.0f, 1.0f,
-50.0f,  -22.0f, 0.0f, 1.0f,
-52.0f,  -22.0f, 0.0f, 1.0f,
//girofar
```

```
-93.0f, -17.0f, 0.0f, 1.0f,
-92.0f, -19.0f, 0.0f, 1.0f,
-88.0f, -19.0f, 0.0f, 1.0f,
-87.0f, -17.0f, 0.0f, 1.0f,
-88.0f, -15.0f, 0.0f, 1.0f,
-92.0f, -15.0f, 0.0f, 1.0f,
//Banda 1
-200.0f, 35.0f, 0.0f, 1.0f,
200.0f, 35.0f, 0.0f, 1.0f,
150.0f, 20.0f, 0.0f, 1.0f,
-150.0f, 20.0f, 0.0f, 1.0f,
//Banda 2
-200.0f, 20.0f, 0.0f, 1.0f,
200.0f, 20.0f, 0.0f, 1.0f,
150.0f, 5.0f, 0.0f, 1.0f,
-150.0f, 5.0f, 0.0f, 1.0f,
//Camp1
-200.0f, 35.0f, 0.0f, 1.0f,
200.0f, 35.0f, 0.0f, 1.0f,
150.0f, 200.0f, 0.0f, 1.0f,
-150.0f, 200.0f, 0.0f, 1.0f,
//Camp2
-200.0f, 5.0f, 0.0f, 1.0f,
200.0f, 5.0f, 0.0f, 1.0f,
150.0f, -200.0f, 0.0f, 1.0f,
-150.0f, -200.0f, 0.0f, 1.0f,
};

GLfloat Colors[] = {
// albastra + roti
  1.0f, 0.0f, 0.0f, 1.0f,
  0.0f, 1.0f, 0.0f, 1.0f,
  0.0f, 0.0f, 1.0f, 1.0f,
  0.0f, 1.0f, 0.0f, 1.0f,
  0.0f, 1.0f, 0.0f, 1.0f,
  0.0f, 0.0f, 1.0f, 1.0f,
  //
  0.0f, 0.0f, 0.0f, 1.0f,
  0.0f, 0.0f, 0.0f, 1.0f,
```

```
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        //
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        // rosie + roti
        1.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 1.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 1.0f, 1.0f,
        0.0f, 1.0f, 0.0f, 1.0f,
        0.0f, 1.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 1.0f, 1.0f,
        //
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        //
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 0.0f, 1.0f,
        // verde + roti
        1.0f, 0.0f, 0.0f, 1.0f,
        1.0f, 0.0f, 0.0f, 1.0f,
        0.0f, 1.0f, 0.0f, 1.0f,
        0.0f, 0.0f, 1.0f, 1.0f,
        0.0f, 0.0f, 1.0f, 1.0f,
        0.0f, 0.0f, 1.0f, 1.0f,
        //
        0.0f, 0.0f, 0.0f, 1.0f,
0.0f, 0.0f, 0.0f, 1.0f,
0.0f, 0.0f, 0.0f, 1.0f,
0.0f, 0.0f, 0.0f, 1.0f,
//
0.0f, 0.0f, 0.0f, 1.0f,
0.0f, 0.0f, 0.0f, 1.0f,
```

```
0.0f, 0.0f, 0.0f, 1.0f,
0.0f, 0.0f, 0.0f, 1.0f,
      //girofar
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      //banda1
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      //banda2
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      //camp1
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      //camp2
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
      1.0f, 0.0f, 0.0f, 1.0f,
      0.0f, 0.0f, 1.0f, 1.0f,
    };

    glGenBuffers(1, &VboId);
    glBindBuffer(GL_ARRAY_BUFFER, VboId);
    glBufferData(GL_ARRAY_BUFFER, sizeof(Vertices), Vertices, GL_STATIC_DRAW);

    glGenVertexArrays(1, &VaoId);
    glBindVertexArray(VaoId);
    glEnableVertexAttribArray(0);
    glVertexAttribPointer(0, 4, GL_FLOAT, GL_FALSE, 0, 0);
```

```
	glGenBuffers(1, &ColorBufferId);
	glBindBuffer(GL_ARRAY_BUFFER, ColorBufferId);
	glBufferData(GL_ARRAY_BUFFER, sizeof(Colors), Colors, GL_STATIC_DRAW);
	glEnableVertexAttribArray(1);
	glVertexAttribPointer(1, 4, GL_FLOAT, GL_FALSE, 0, 0);
}
void DestroyVBO(void)
{
	glDisableVertexAttribArray(1);
	glDisableVertexAttribArray(0);
	glBindBuffer(GL_ARRAY_BUFFER, 0);
	glDeleteBuffers(1, &ColorBufferId);
	glDeleteBuffers(1, &VboId);
	glBindVertexArray(0);
	glDeleteVertexArrays(1, &VaoId);
}

void CreateShaders(void)
{
	ProgramId = LoadShaders("Proiect01_Shader.vert", "Proiect01_Shader.frag");
	glUseProgram(ProgramId);
}
void DestroyShaders(void)
{
	glDeleteProgram(ProgramId);
}

void Initialize(void)
{
	glClearColor(1.0f, 1.0f, 1.0f, 0.0f);
	CreateVBO();
	CreateShaders();
	i = -40.0f;
	j = -50.0f;
	ii = -40.0f;
	jj = -50.0f;
	codColLocation = glGetUniformLocation(ProgramId, "codCuloare");
	myMatrixLocation = glGetUniformLocation(ProgramId, "myMatrix");
```

```
}
void RenderFunction(void)
{
	glClear(GL_COLOR_BUFFER_BIT);

	resizeMatrix = glm::ortho(-width, width, -height, height);
	matrTransl = glm::translate(glm::mat4(1.0f), glm::vec3(i + 40.0f, j + 80.0f, 0.0));
	matrTrans2 = glm::translate(glm::mat4(1.0f), glm::vec3(ii + 40.0f, jj + 80.0f, 0.0));
	matrRot = glm::rotate(glm::mat4(1.0f), angle, glm::vec3(0.0, 0.0, 1.0));
	matrDepl1 = glm::translate(glm::mat4(1.0f), glm::vec3(90.0f, 17.0f, 0.0f));
	matrDepl2 = glm::translate(glm::mat4(1.0f), glm::vec3(-90.0f, -17.0f, 0.0f));

	//Banda1
	myMatrix = resizeMatrix;
	codCol = 4;
	glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
	glUniform1i(codColLocation, codCol);
	glDrawArrays(GL_POLYGON, 48, 4);
	//Banda2
	myMatrix = resizeMatrix;
	codCol = 5;
	glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
	glUniform1i(codColLocation, codCol);
	glDrawArrays(GL_POLYGON, 52, 4);
	//Camp1
	myMatrix = resizeMatrix;
	codCol = 6;
	glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
	glUniform1i(codColLocation, codCol);
	glDrawArrays(GL_POLYGON, 56, 4);
	//Camp2
	myMatrix = resizeMatrix;
	codCol = 7;
	glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
	glUniform1i(codColLocation, codCol);
	glDrawArrays(GL_POLYGON, 60, 4);
	//Masina albastra
	myMatrix = resizeMatrix * matrTrans2;
	codCol = 3;
```

```
glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
glUniform1i(codColLocation, codCol);
glDrawArrays(GL_POLYGON, 0, 6);
//Roata A1
myMatrix = resizeMatrix * matrTrans2;
codCol = 0;
glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
glUniform1i(codColLocation, codCol);
glDrawArrays(GL_POLYGON, 6, 4);
//Roata R1
myMatrix = resizeMatrix * matrTrans2;
codCol = 0;
glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
glUniform1i(codColLocation, codCol);
glDrawArrays(GL_POLYGON, 10, 4);
//Masina rosie
myMatrix = resizeMatrix * matrTransl;
codCol = 1;
glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
glUniform1i(codColLocation, codCol);
glDrawArrays(GL_POLYGON, 14, 6);
//Roata R1
myMatrix = resizeMatrix * matrTransl;
codCol = 0;
glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
glUniform1i(codColLocation, codCol);
glDrawArrays(GL_POLYGON, 20, 4);
//Roata R2
myMatrix = resizeMatrix * matrTransl;
codCol = 0;
glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
glUniform1i(codColLocation, codCol);
glDrawArrays(GL_POLYGON, 24, 4);
//Masina verde
myMatrix = resizeMatrix * matrTrans2;
codCol = 2;
glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
glUniform1i(codColLocation, codCol);
glDrawArrays(GL_POLYGON, 28, 6);
```

```
	//Roata V1
	myMatrix = resizeMatrix * matrTrans2;
	codCol = 0;
	glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
	glUniform1i(codColLocation, codCol);
	glDrawArrays(GL_POLYGON, 34, 4);
	//Roata V2
	myMatrix = resizeMatrix * matrTrans2;
	codCol = 0;
	glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
	glUniform1i(codColLocation, codCol);
	glDrawArrays(GL_POLYGON, 38, 4);
	//Girofar
	myMatrix = resizeMatrix * matrTrans2 * matrDepl2 * matrRot * matrDepl1;
	codCol = 0;
	glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
	glUniform1i(codColLocation, codCol);
	glDrawArrays(GL_POLYGON, 42, 6);
	glutSwapBuffers();
	glFlush();
}
void Cleanup(void)
{
	DestroyShaders();
	DestroyVBO();
}

int main(int argc, char* argv[])
{
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
	glutInitWindowPosition(100, 100);
	glutInitWindowSize(800, 800);
	glutCreateWindow("Careful drivers");
	glewInit();
	Initialize();
	glutDisplayFunc(RenderFunction);
	glutIdleFunc(move);
	glutCloseFunc(Cleanup);
```

```
	glutMainLoop();
}
```

2) Shadere

Proiect01_Shader.frag

```
#version 330

in vec4 ex_Color;
uniform int codCuloare;

out vec4 out_Color;

void main(void)
{
  switch (codCuloare)
  {
	case 0:
	  out_Color = ex_Color;
	  break;
	case 1:
		out_Color=vec4 (1.0, 0.0, 0.0, 0.0);
		break;
	case 2:
		out_Color=vec4 (0.0, 0.5, 0.0, 0.0);
		break;
	case 3:
		out_Color=vec4 (0.0, 0.0, 0.8, 0.0);
		break;
	case 4:
		out_Color=vec4 (0.7, 0.4, 0.1, 0.0);
		break;
	case 5:
		out_Color=vec4 (0.4, 0.3, 0.1, 0.0);
		break;
	case 6:
		out_Color=vec4 (0.0, 0.2, 0.1, 0.0);
		break;
	case 7:
		out_Color=vec4 (0.6, 0.9, 0.6, 0.0);
		break;
	default:
		break;
  };
}
```

Proiect01_Shader.vert

```
#version 330

layout (location = 0) in vec4 in_Position;
layout (location = 1) in vec4 in_Color;

out vec4 gl_Position;
out vec4 ex_Color;
uniform mat4 myMatrix;
```

```
void main(void)
{
    gl_Position = myMatrix*in_Position;
    ex_Color = in_Color;
}
```