Documentatie Proiect 1

--Fugi de Politie--

Proiect realizat de:

Buturuga George-Alexandru

Ionita Roxana Diana

Ghita Filip Darius

Alexandrescu Marian Gabriel

Grupa 364

Universitatea din Bucuresti

Contents

[Tema proiectului 2](#_Toc133958770)

[Descriere succinta 3](#_Toc133958771)

[Primitive, transformari, input interactiv 3](#_Toc133958772)

[Originalitate 3](#_Toc133958773)

[Contributii 3](#_Toc133958774)

[Resurse utilizate: 3](#_Toc133958775)

[Functionalitate 3](#_Toc133958776)

[Meniu start/control 3](#_Toc133958777)

[Functia Display() 4](#_Toc133958778)

[Meniu muzica: 4](#_Toc133958779)

[Keys 5](#_Toc133958780)

[Functia startgame() 6](#_Toc133958781)

[Functia MiscareGirofar() 6](#_Toc133958782)

[Pauza: 6](#_Toc133958783)

[Link video – demo 7](#_Toc133958784)

[Link GitHub 7](#_Toc133958785)

[Cod Sursa 7](#_Toc133958785)

## Tema proiectului

Tema proiectului nostru este reprezentata de un joc cu masini, in care un infractor este urmarit de politie. Am folosit ca baza, proiectul cu numarul 5 (schelet\_proiect5.txt).

## Descriere succinta

Pe baza scheletului, am adaugat functionalitati noi, precum muzica de fundal si meniu ce permite schimbarea acesteia. De asemenea, am adaugat optiunea de pausa si resume, dar am crescut si dificultatea jocului prin introducerea de obstacole odata ce scorul atinge un prag.

## Primitive, transformari, input interactiv

Am folosit puncte, drepte si triunghiuri pentru a desena cele doua masini si decorul, Spre exemplu, punctele si dreptele sunt folosite la desenarea dreptunghiurilor ce reprezinta sasiul masinilor, triunghiul este folosit pt desenarea girofarului si la coroana copacilor. Ele se pot observa si in functiile drawHouse(), drawTree(), drawStone() si drawBarrier().

In proiectul nostru, compunerea transformarilor se realizeaza prin utilizarea functiei transform, a functiei glPushMatrix() si glPopMatrix();

Miscam obiectele cu ajutorul functiei glTranslated. De asemenea folosim si axele X si Y pentru alte obiecte cum ar fi copacii si barierele.

Girofarul se roteste cu ajutorul functiei GlRotated().

Inputul interactiv este implementat prin functia Mouse(), cu ajutorul careia utilizatorul selecteaza optiunea de start sau control, dar si prin meniul de schimbare al melodiei de fundal.

## Originalitate

Consideram ca proiectul nostru este unul original deoarece am folosit functii speciale pentru introducerea de fisiere wav (muzica). In plus, am facut si un submeniu in care utilizatorul o poate schimba. Am creeat obiecte de fundal, ce se misca scalabil cu acceleratia masinii, de asemenea, am adaugat si un obiect nou(bariera). In urma coliziunii cu aceasta se sfarseste jocul.

## Contributii

Marian – grafica, desene, optimizare buguri, idei

Darius – meniuri, semnalizare, scor, idei

George – meniu schimbare muzica si muzica fundal, miscare, documentatie

Roxana – optiune pauza, documentatie si meniu start/control

## Resurse utilizate:

-laboratoare

-schelet 5

-youtube

-chatgpt pt anumite erori

## Functionalitate

### Meniu start/control

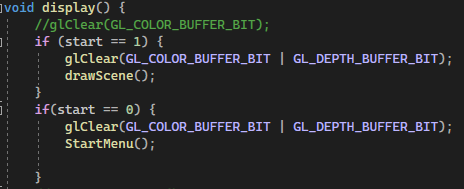
Jocul afiseaza initial un meniu ce cuprinde 2 butoane, unul de start si unul de control, ambele fiind gestionate de functia mouse()

Butonul de start functioneaza prin intermediul variabilei locale start, ce retine starea jocului, daca acesta a fost pornit sau nu.

Meniul este implementat in functia startMenu(), in care ne-am ocupat de grafica interfetei de start (am desenat peisajul, copacii si chenarul cu cele 2 butoane), iar apoi am apelat functia ControlMenu(), pentru a afisa in partea de sus instructiunile jocului si in partea de jos, comenzile, atunci cand utilizatorul solocita acest lucru (click pe buton Control, ce modifica in functia mouse(), variabila globala control.)

### Functia Display()

Scenele jocului sunt gestionate de functia display().

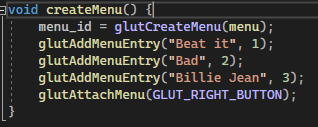


Atunci cand jocul inca nu a pornit, deci variabila start este 0, se va randa pagina de start, cu peisajul si meniul cu butoane. Cand start este 1, jocul este pornit si este randata functia DrawScene(), in care am introdus si o modalitate de a afisa scorul , actualizat de fiecare data cand hotul trece de masina politistului fara sa fie prins. Afisarea lui in scena jocului se face prin functia RenderString.

### Meniu muzica:

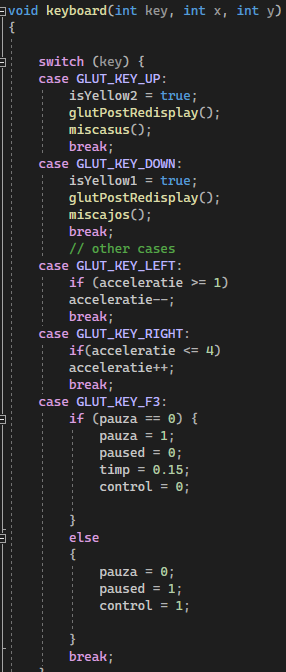
Pentru redarea muzicii am folosit comanda PlaySound. Pentru cele 3 elemente din acest meniu, am utilizat un switch in functia menu(), ce are ca scop modificarea caii spre fisierul nostru, astfel incat sa fie deschisa melodia selectata.

Codul urmator a fost utilizat pentru integrarea meniului in proiect, functia createMenu() este apelata in program.



### Keys

Am utilizat un switch



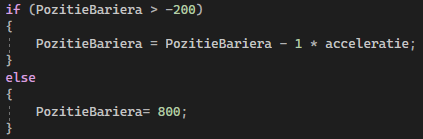
Sagetile up si down muta masinuta hotului pe banda de deasupra, respectiv dedesubtul celei curente. Masina Hotului scimba benzile cu ajutorul variabilei j, ce trece pe rand la anumite valori: banda 0 = 0, banda 1 = 160, banda 2 = 320.

Politistul schimba pozitia random, poziitile fiind aceleasi, la fel si celelalte obstacole(bariera). Hotul este prins atunci cand pozitia sa se intersecteaza cu unul dintre obiectele ce vin din sens opus.

Sageata dreapta schimba viteza jocului, iar tasta F3 este folosita pentru a pune pauza si a iesi din pauza in cazul in care aceasta este pornita.

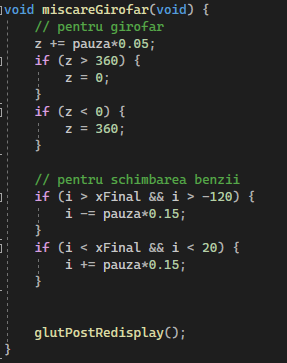
### Functia startgame()

Functia contine conditia de gameover, cand hotul se intersecteaza cu un obstacol, ok = 0. Tot aici este efectuata miscarea obiectelor din cadru, acest lucru fiind scalat cu acceleratia. De asemenea, pentru a da impresia de continuitate, cand obiectul iese din cadru, acesta se va reintoarce in pozitia initiala, in cazul de fata:800. Exemplu:



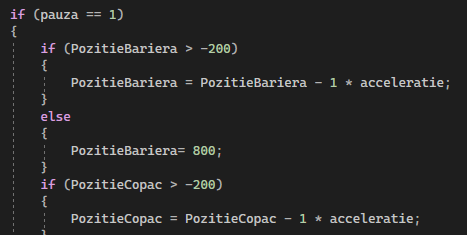
### Functia MiscareGirofar()

Functia gestioneaza miscarea girofarului, prin folosirea variabilei globale pauza. In timpul pauzei de la joc, girofarul nu se mai misca, variabila z devenind dependenta de variabila pauza.



### Pauza:

Atunci cand programul trece in modul pauza, acesta va opri orice miscare vizibila pe ecran. Unele operatii vor primi operatorul 0 ce va face operatiile nule ceea ce va duce la oprirea miscarii (exemplu de mai sus cu girofarul) iar altele contin un if ce va opri executia operatilor. Exemplu:



## Link video – demo

<https://drive.google.com/file/d/1LQ3oArKwQG22LsihqXlOG2MJSx2qqWpO/view?usp=sharing>

## Link GitHub

<https://github.com/alexbuturuga/Proiect-Grafica---Fugi-de-Politie.git>

Cod sursa:

(Fisier 1)

#include <iostream>

#include <windows.h>

#include <GL/glut.h>

#include <GL/freeglut.h>

#include <Windows.h>

#include <filesystem>

#include "georg.cpp"

#include <string>

#include "maur.cpp"

#include "darius.cpp"

#include "start.cpp"

using namespace std;

double xFinal = 20;

std::wstring directory;

GLdouble left\_m = -100.0;

GLdouble right\_m = 700.0;

GLdouble bottom\_m = -140.0;

GLdouble top\_m = 460.0;

int exponent = 0;

int acceleratie = pow(2,exponent);

double ok = 1;

int misca = 0;

double j = 0.0;

double i = 0.0;

double z = 0.0;

double PozitieCopac = 175;

double PozitieRock = 400;

double PozitieCasa = 660;

double PozitieBoschet = 900;

int PozitieBariera = 800;

bool isYellow1 = false;

bool isYellow2 = false;

double contor = 0;

double loc\_vert = 800;

int vector[3] = { 0, 160, 320 };

double height = vector[rand() % 3];

int score = 0;

double timp = 0.15;

int pct = 1000;

int ok2 = 0;

double rsj, rdj, rss, rds = 0;

double paused = 0;

bool gameStarted = false;

int pauza = 1;

int menu\_id;

int start = 0;

int control = 0;

int r = 10;

const int font1 = (int)GLUT\_BITMAP\_TIMES\_ROMAN\_24;

const int font2 = (int)GLUT\_BITMAP\_HELVETICA\_18;

const int font3 = (int)GLUT\_BITMAP\_8\_BY\_13;

int randomNumber(int min, int max) {

return rand() % (max - min + 1) + min;

}

void RenderString(float x, float y, void\* font, const unsigned char\* string)

{

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

glRasterPos2f(x, y);

glutBitmapString(font, string);

}

void init(void)

{

glClearColor(0.5, 0.5, 0.5, 0.0);

glMatrixMode(GL\_PROJECTION);

glOrtho(left\_m, right\_m, bottom\_m, top\_m, -1.0, 1.0);

}

void firstDesign() {

glColor3f(0.55, 0.788, 0.451);

// Iarba de jos

glBegin(GL\_POLYGON);

glVertex2i(-100, -140);// Stanga jos

glVertex2i(700, -140); // Dreapta jos

glVertex2i(700, -80); // Dreapta sus

glVertex2i(-100, -80); // Stanga sus

glEnd();

// Iarba de sus

glBegin(GL\_POLYGON);

glVertex2i(-100, 400);// Stanga jos

glVertex2i(700, 400); // Dreapta jos

glVertex2i(700, 460); // Dreapta sus

glVertex2i(-100, 460);// Stanga sus

glEnd();

}

void miscareGirofar(void) {

// pentru girofar

z += pauza\*0.05;

if (z > 360) {

z = 0;

}

if (z < 0) {

z = 360;

}

// pentru schimbarea benzii

if (i > xFinal && i > -120) {

i -= pauza\*0.15;

}

if (i < xFinal && i < 20) {

i += pauza\*0.15;

}

glutPostRedisplay();

}

void startgame(void)

{

if ((height != j || (loc\_vert > 10 || loc\_vert < -100)) && (ok == 1))

{

if (160 \* r == j && !(PozitieBariera > 60 || PozitieBariera < 15))

{

ok = 0;

pauza = 0;

}

if (pauza == 1)

{

if (PozitieBariera > -200)

{

PozitieBariera = PozitieBariera - 1 \* acceleratie;

}

else

{

PozitieBariera= 800;

}

if (PozitieCopac > -200)

{

PozitieCopac = PozitieCopac - 1 \* acceleratie;

}

else

{

PozitieCopac = 800;

}

if (PozitieCasa > -200)

{

PozitieCasa = PozitieCasa - 1 \* acceleratie;

}

else

{

PozitieCasa = 800;

}

if (PozitieBoschet > -200)

{

PozitieBoschet = PozitieBoschet - 1 \* acceleratie;

}

else

{

PozitieBoschet = 800;

}

if (PozitieRock > -200)

{

PozitieRock = PozitieRock - 1 \* acceleratie;

}

else

{

PozitieRock = 800;

}

if (i < -380)

{

i = 0;

}

i = i - 12 \* timp + 1;

loc\_vert -= 6 \* timp + 1 \* acceleratie;

if (loc\_vert < -350)

{

score += 100;

height = vector[rand() % 3];

cout << "Score: " << score << endl;

loc\_vert = 700;

}

if (score >= pct && pct <= 15000)

{

timp += 0.1;

pct += 1000;

}

glutPostRedisplay();

}

}

else {

ok = 0;

pauza = 0;

}

}

void renderPauseMenu() {

glPushMatrix();

glLoadIdentity();

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

glRasterPos2f(240.0f, 200.0f);

glutBitmapString(GLUT\_BITMAP\_TIMES\_ROMAN\_24, (unsigned const char\*)"Paused game");

glRasterPos2f(210.0f, 170.0f);

glutBitmapString(GLUT\_BITMAP\_HELVETICA\_18, (unsigned const char\*)"Press 'Right' to Resume");

glPopMatrix();

}

void timer(int value) {

isYellow1 = false;

isYellow2 = false;

glutPostRedisplay();

glutTimerFunc(500, timer, 0); // call this function again after 1000 milliseconds

}

void mouse(int button, int state, int x, int y) {

if (start == 0)

{

//Controls

if (button == GLUT\_LEFT\_BUTTON && state == GLUT\_DOWN && x >= 450 && x <= 550 && y >= 300 && y <= 330) {

if (control == 0)

{

control = 1;

}

else

{

control = 0;

}

}

//Start

if (button == GLUT\_LEFT\_BUTTON && state == GLUT\_DOWN && x >= 250 && x <= 350 && y >= 300 && y <= 330) {

start = 1;

}

}

RenderString(177.0f, 140.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Start");

}

void ControlMenu()

{

glColor3f(0.7, 0.7, 0.7);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1, 0x00FF);

glBegin(GL\_POLYGON);

glVertex2i(500, 250);

glVertex2i(100, 250);

glVertex2i(100, 410);

glVertex2i(500, 410);

glEnd();

glDisable(GL\_LINE\_STIPPLE);

RenderString(240.0f, 380.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Instructiuni :");

RenderString(110.0f, 350.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Trebuie sa fii in totalitate pe alta");

RenderString(110.0f, 320.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Banda pentru a nu fi prins de politie.");

RenderString(110.0f, 290.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"De la 1000 de puncte apar obstacole.");

glColor3f(0.7, 0.7, 0.7);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1, 0x00FF);

glBegin(GL\_POLYGON);

glVertex2i(500, -100);

glVertex2i(100, -100);

glVertex2i(100, 90);

glVertex2i(500, 90);

glEnd();

glDisable(GL\_LINE\_STIPPLE);

RenderString(120.0f, -85.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Click dreapta - alege melodiia");

RenderString(120.0f, -55.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"F3 - Start/Stop");

RenderString(120.0f, -25.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Sageata sus - Schimba banda in sus");

RenderString(120.0f, 5.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Sageata jos - Schimba banda in sus");

RenderString(120.0f, 35.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Sageata dreapta - Creste acceleratia");

RenderString(120.0f, 65.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Sageata stanga - Scade acceleratia");

}

void StartMenu(void)

{

glColor3f(0.55, 0.788, 0.451);

// Iarba de jos

glBegin(GL\_POLYGON);

glVertex2i(-100, -140);// Stanga jos

glVertex2i(700, -140); // Dreapta jos

glVertex2i(700, -80); // Dreapta sus

glVertex2i(-100, -80); // Stanga sus

glEnd();

// Iarba de sus

glBegin(GL\_POLYGON);

glVertex2i(-100, 400);// Stanga jos

glVertex2i(700, 400); // Dreapta jos

glVertex2i(700, 460); // Dreapta sus

glVertex2i(-100, 460);// Stanga sus

glEnd();

RenderString(160.0f, 425.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Nu lasa politia sa te prinda!");

//Chenar meniu

glTranslated(i, 0.0, 0.0);

glColor3f(0.7, 0.7, 0.7);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1, 0x00FF);

glBegin(GL\_POLYGON);

glVertex2i(500, 100);

glVertex2i(100, 100);

glVertex2i(100, 200);

glVertex2i(500, 200);

glEnd();

glDisable(GL\_LINE\_STIPPLE);

//Butoane

glTranslated(i, 0.0, 0.0);

glColor3f(0.7, 0.7, 1);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1, 0x00FF);

glBegin(GL\_POLYGON);

glVertex2i(250, 130);

glVertex2i(150, 130);

glVertex2i(150, 170);

glVertex2i(250, 170);

glEnd();

glDisable(GL\_LINE\_STIPPLE);

glColor3f(0.7, 0.7, 1);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1, 0x00FF);

glBegin(GL\_POLYGON);

glVertex2i(470, 130);

glVertex2i(350, 130);

glVertex2i(350, 170);

glVertex2i(470, 170);

glEnd();

glDisable(GL\_LINE\_STIPPLE);

RenderString(177.0f, 140.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Start");

RenderString(375.0f, 140.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Control");

// Delimitare sosea

glLineWidth(3);

glColor3f(1, 1, 1);

// Delimitam soseaua de iarba partea de jos

glBegin(GL\_LINES);

glVertex2i(-100, -80);

glVertex2i(1500, -80);

glEnd();

// Delimitam soseaua de iarba partea de sus

glBegin(GL\_LINES);

glVertex2i(-100, 400);

glVertex2i(1500, 400);

glEnd();

//Desenam copacii

for (int i = 0; i <= 20; i++)

{

drawTree(-40 + 120\*i, -120);

}

//control

if (control == 1)

{

ControlMenu();

}

// Liniile intrerupte

glTranslated(i, 0.0, 0.0);

glColor3f(1, 1, 1);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1, 0x00FF);

glBegin(GL\_LINES);

glVertex2i(-100, 80);

glVertex2i(1500, 80);

glEnd();

glDisable(GL\_LINE\_STIPPLE);

glColor3f(1, 1, 1);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1, 0x00FF);

glBegin(GL\_LINES);

glTranslated(i, 0.0, 0.0);

glVertex2i(-100, 240);

glVertex2i(1500, 240);

glEnd();

glDisable(GL\_LINE\_STIPPLE);

glPopMatrix();

glPushMatrix();

//Afisare scor

//string scoreText = "Score: " + to\_string(score); // convert the score to a string

//const unsigned char\* scoreStr = reinterpret\_cast<const unsigned char\*>(scoreText.c\_str()); // convert the string to a char array

//RenderString(580.0f, 425.0f, GLUT\_BITMAP\_HELVETICA\_18, scoreStr); // display the score

glPopMatrix();

glutPostRedisplay();

glutSwapBuffers();

glFlush();

}

void drawScene(void)

{

if (start == 1)

{

glColor3f(0.55, 0.788, 0.451);

// Iarba de jos

glBegin(GL\_POLYGON);

glVertex2i(-100, -140);// Stanga jos

glVertex2i(700, -140); // Dreapta jos

glVertex2i(700, -80); // Dreapta sus

glVertex2i(-100, -80); // Stanga sus

glEnd();

// Iarba de sus

glBegin(GL\_POLYGON);

glVertex2i(-100, 400);// Stanga jos

glVertex2i(700, 400); // Dreapta jos

glVertex2i(700, 460); // Dreapta sus

glVertex2i(-100, 460);// Stanga sus

glEnd();

RenderString(160.0f, 425.0f, GLUT\_BITMAP\_TIMES\_ROMAN\_24, (const unsigned char\*)"Nu lasa politia sa te prinda!");

// Delimitare sosea

glLineWidth(3);

glColor3f(1, 1, 1);

// Delimitam soseaua de iarba partea de jos

glBegin(GL\_LINES);

glVertex2i(-100, -80);

glVertex2i(1500, -80);

glEnd();

// Delimitam soseaua de iarba partea de sus

glBegin(GL\_LINES);

glVertex2i(-100, 400);

glVertex2i(1500, 400);

glEnd();

glPushMatrix();

// Liniile intrerupte

//Dresenam Obiecte

drawTree(PozitieCopac, -130);

drawBush(PozitieBoschet, -120);

drawHouse(PozitieCasa, -130);

drawRock(PozitieRock, -130);

glTranslated(i\*acceleratie, 0.0, 0.0);

glColor3f(1, 1, 1);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1, 0x00FF);

glBegin(GL\_LINES);

glVertex2i(-100, 80);

glVertex2i(3500, 80);

glEnd();

glDisable(GL\_LINE\_STIPPLE);

glColor3f(1, 1, 1);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1, 0x00FF);

glBegin(GL\_LINES);

glTranslated(i \* acceleratie, 0.0, 0.0);

glVertex2i(-100, 240);

glVertex2i(3500, 240);

glEnd();

glDisable(GL\_LINE\_STIPPLE);

glPopMatrix();

glPushMatrix();

//Afisare scor

string scoreText = "Score: " + to\_string(score); // convert the score to a string

const unsigned char\* scoreStr = reinterpret\_cast<const unsigned char\*>(scoreText.c\_str()); // convert the string to a char array

RenderString(580.0f, 425.0f, GLUT\_BITMAP\_HELVETICA\_18, scoreStr); // display the score

//desenam masina

glPushMatrix();

glTranslated(0.0, j, 0.0);

//masina lu' Hotzu

glColor3f(0.996, 0.365, 0.149);

glRecti(-45, -25, -20, 25);

glColor3f(0, 0, 0);

glRecti(-20, -25, 45, 25);

glColor3f(0.996, 0.365, 0.149);

glRecti(45, -25, 70, 25);

//faruri

glColor3f(0.8, 0.5, 0);

glRecti(70, -25, 80, -10);

glColor3f(0.8, 0.5, 0);

glRecti(70, 10, 80, 25);

//stopuri

glColor3f(1, 0, 0);

glRecti(-55, -25, -45, -10);

glColor3f(1, 0, 0);

glRecti(-55, 10, -45, 25);

if (isYellow1) {

glColor3f(1.0, 1.0, 0.0); // yellow color

}

else {

glColor3f(0.8, 0.5, 0.0); // original color

}

glRecti(70, -25, 80, -10);

if (isYellow2) {

glColor3f(1.0, 1.0, 0.0); // yellow color

}

else {

glColor3f(0.8, 0.5, 0.0); // original color

}

glRecti(70, 10, 80, 25);

if (ok == 0)

{

rsj = 8;

rss = -8;

rdj = -8;

rds = 8;

}

if (paused == 1) {

timp = 0;

renderPauseMenu();

}

if (control == 1)

{

glPopMatrix();

glPushMatrix();

ControlMenu();

}

//ok2 verifica daca este pornita muzica de final de joc

if (ok == 0 && ok2 == 0)

{

std::wstring soundPath = directory + L"\\gameover.wav";

if (PlaySound(soundPath.c\_str(), NULL, SND\_ASYNC))

cout << "Muzica Joc Terminat!";

ok2 = 1;

}

glPopMatrix();

glPopMatrix();

if (ok == 0) {

RenderString(250.0f, 200.0f, GLUT\_BITMAP\_8\_BY\_13, (const unsigned char\*)"GAME OVER, AI FOST PRINS!!");

}

if (contor == 1 && (j != 160 && j != 320))

{

j = j + 2 \* pauza;

}

else if (contor == -1 && (j != 160 && j != 0))

{

j = j - 2 \* pauza;

}

else {

contor = 0;

misca = 0;

}

//Girofar

if (score >= 100)

{

if (r == 10)

{

r = 1;

PozitieBariera = 800;

}

//glTranslated(loc\_vert, height - 48, 0.0);

if(PozitieBariera == -199)

r = randomNumber(0, 2);

drawBarrier(PozitieBariera, r\*150-40);

}

//MASINA DE POLITIE

//desenam a doua masina (adversara)

glPushMatrix();

glTranslated(loc\_vert, height - 48, 0.0);

//glColor3f(0.471, 0.667, 0.949);

//glRecti(-45, -15, 45, 15);

//SASIU

glRotated(90, 0, 0, 1);

glColor3f(0, 0, 0);

glRectf(20, -180, 80, -155);

glColor3f(1, 1, 1);

glRectf(20, -155, 80, -115);

glColor3f(0, 0, 0);

glRectf(20, -115, 80, -90);

//STOPURI

glColor3f(1, 0, 0);

glRectf(25, -185, 35, -180);

glColor3f(1, 0, 0);

glRectf(65, -185, 75, -180);

//FARURI

glColor3f(0.8, 0.5, 0);

glRectf(25, -90, 35, -85);

glColor3f(0.8, 0.5, 0);

glRectf(65, -90, 75, -85);

glColor3f(1, 1, 0);

glBegin(GL\_TRIANGLES);

glVertex2f(30, -85);

glVertex2f(0, 0);

glVertex2f(60, 0);

glVertex2f(40, 0);

glVertex2f(100, 0);

glVertex2f(70, -85);

glEnd();

// GIROFAR

glTranslated(50, -135, 0);

glRotated(z\*20, 0, 0, 1);

glTranslated(-50, 135, 0);

//girofar

glColor4f(1, 0, 0, 0.5);

glBegin(GL\_TRIANGLES);

glVertex2f(10, -115);

glVertex2f(50, -135);

glVertex2f(10, -155);

glEnd();

glColor4f(0, 0, 1, 0.5);

glBegin(GL\_TRIANGLES);

glVertex2f(50, -135);

glVertex2f(90, -115);

glVertex2f(90, -155);

glEnd();

glPopMatrix();

glPopMatrix();

// incrementare pentru girofar si alte translatii

glutIdleFunc(miscareGirofar);

glPopMatrix();

startgame();

glutPostRedisplay();

glutSwapBuffers();

glFlush();

}

}

void reshape(int w, int h)

{

glViewport(0, 0, (GLsizei)w, (GLsizei)h);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(-100.0, 700.0, -140.0, 460.0, -1.0, 1.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void miscasus(void)

{

if (ok != 0 && pauza)

{

if (j < 320)

{

contor = 1;

j += 2;

misca = 1;

}

glutPostRedisplay();

}

}

void miscajos(void)

{

if (ok != 0 && pauza)

{

if (j > 0)

{

contor = -1;

j -= 2;

misca = -1;

}

glutPostRedisplay();

}

}

void display() {

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

if (start == 1) {

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

drawScene();

}

if(start == 0) {

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

StartMenu();

}

//glutSwapBuffers();

}

void menu(int choice) {

wchar\_t path[MAX\_PATH];

GetModuleFileNameW(NULL, path, MAX\_PATH);

directory = std::wstring(path);

directory = directory.substr(0, directory.find\_last\_of(L"\\/"));

std::wstring soundPath;

soundPath = directory + L"\\muzica.wav";

std::wcout << L"Project directory: " << soundPath << std::endl;

switch (choice) {

case 1:

//Melodie 1

soundPath = directory + L"\\muzica.wav";

break;

case 2:

soundPath = directory + L"\\muzica2.wav";

break;

case 3:

soundPath = directory + L"\\muzica3.wav";

break;

}

if (PlaySound(soundPath.c\_str(), NULL, SND\_ASYNC))

cout << "Muzica 1";

glutPostRedisplay();

}

void createMenu() {

menu\_id = glutCreateMenu(menu);

glutAddMenuEntry("Beat it", 1);

glutAddMenuEntry("Bad", 2);

glutAddMenuEntry("Billie Jean", 3);

glutAttachMenu(GLUT\_RIGHT\_BUTTON);

}

void keyboard(int key, int x, int y)

{

switch (key) {

case GLUT\_KEY\_UP:

isYellow2 = true;

glutPostRedisplay();

miscasus();

break;

case GLUT\_KEY\_DOWN:

isYellow1 = true;

glutPostRedisplay();

miscajos();

break;

// other cases

case GLUT\_KEY\_LEFT:

if (acceleratie >= 1)

acceleratie--;

break;

case GLUT\_KEY\_RIGHT:

if(acceleratie <= 4)

acceleratie++;

break;

case GLUT\_KEY\_F3:

if (pauza == 0) {

pauza = 1;

paused = 0;

timp = 0.15;

control = 0;

}

else

{

pauza = 0;

paused = 1;

control = 1;

}

break;

}

}

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

{

wchar\_t path[MAX\_PATH];

GetModuleFileNameW(NULL, path, MAX\_PATH);

directory = std::wstring(path);

directory = directory.substr(0, directory.find\_last\_of(L"\\/"));

std::wstring soundPath = directory + L"\\muzica.wav";

std::wcout << L"Project directory: " << soundPath << std::endl;

if (PlaySound(soundPath.c\_str(), NULL, SND\_ASYNC))

cout << "Muzica fundal";

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowSize(800, 600);

glutInitWindowPosition(100, 100);

glutCreateWindow("Depaseste masinile - mini game");

glutMouseFunc(mouse);

init();

glutDisplayFunc(display);

glutReshapeFunc(reshape);

glutSpecialFunc(keyboard);

glutTimerFunc(1000, timer, 0);

createMenu();

glutMainLoop();

}

(Fisier 2)

#include <iostream>

#include <windows.h>

#include <GL/freeglut.h>

#include <Windows.h>

#include <filesystem>

#include <string>

void drawTree(GLdouble x, GLdouble y)

{

// Draw the trunk

glColor3f(0.43, 0.16, 0.0);

glBegin(GL\_POLYGON);

glVertex2d(x, y);

glVertex2d(x + 10, y);

glVertex2d(x + 10, y + 50);

glVertex2d(x, y + 50);

glEnd();

// Draw the foliage

glColor3f(0.0, 0.5, 0.0);

glBegin(GL\_POLYGON);

glVertex2d(x - 20, y + 50);

glVertex2d(x + 30, y + 50);

glVertex2d(x + 5, y + 90);

glEnd();

}

void drawRock(GLdouble x, GLdouble y)

{

// Draw the rock

glColor3f(0.5, 0.5, 0.5); // Set the color to gray

glBegin(GL\_POLYGON);

glVertex2d(x, y);

glVertex2d(x + 30, y);

glVertex2d(x + 20, y + 20);

glVertex2d(x + 30, y + 30);

glVertex2d(x + 10, y + 50);

glVertex2d(x - 10, y + 30);

glVertex2d(x, y + 20);

glEnd();

}

void drawBush(GLdouble x, GLdouble y)

{

// Draw the bush

glColor3f(0.0, 0.5, 0.0); // Set the color to green

glBegin(GL\_POLYGON);

glVertex2d(x, y);

glVertex2d(x + 20, y);

glVertex2d(x + 30, y + 10);

glVertex2d(x + 40, y + 10);

glVertex2d(x + 50, y);

glVertex2d(x + 70, y);

glVertex2d(x + 70, y + 20);

glVertex2d(x + 60, y + 30);

glVertex2d(x + 50, y + 40);

glVertex2d(x + 40, y + 50);

glVertex2d(x + 30, y + 50);

glVertex2d(x + 20, y + 40);

glVertex2d(x + 10, y + 30);

glVertex2d(x, y + 20);

glEnd();

}

void drawHouse(GLdouble x, GLdouble y)

{

// Draw the walls

glColor3f(0.7, 0.7, 0.7); // Set the color to light gray

glBegin(GL\_POLYGON);

glVertex2d(x, y);

glVertex2d(x + 80, y);

glVertex2d(x + 80, y + 60);

glVertex2d(x, y + 60);

glEnd();

// Draw the roof

glColor3f(0.8, 0.2, 0.2); // Set the color to red

glBegin(GL\_TRIANGLES);

glVertex2d(x, y + 60);

glVertex2d(x + 40, y + 90);

glVertex2d(x + 80, y + 60);

glEnd();

// Draw the door

glColor3f(0.5, 0.3, 0.0); // Set the color to brown

glBegin(GL\_POLYGON);

glVertex2d(x + 30, y);

glVertex2d(x + 50, y);

glVertex2d(x + 50, y + 40);

glVertex2d(x + 30, y + 40);

glEnd();

// Draw the window

glColor3f(0.5, 0.5, 1.0); // Set the color to light blue

glBegin(GL\_POLYGON);

glVertex2d(x + 60, y + 20);

glVertex2d(x + 70, y + 20);

glVertex2d(x + 70, y + 30);

glVertex2d(x + 60, y + 30);

glEnd();

}

(Fisier 3)

#include <iostream>

#include <windows.h>

#include <GL/freeglut.h>

#include <Windows.h>

#include <filesystem>

#include <string>

void drawBarrier(GLdouble x, GLdouble y)

{

// Draw the white zone

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

glVertex2d(x, y);

glVertex2d(x + 20, y);

glVertex2d(x + 20, y + 100);

glVertex2d(x, y + 100);

glEnd();

//Draw the red bars

glColor3f(1.0, 0.0, 0.0);

glBegin(GL\_POLYGON);

glVertex2d(x, y);

glVertex2d(x + 20, y);

glVertex2d(x + 20, y + 20);

glEnd();

glBegin(GL\_POLYGON);

glVertex2d(x, y + 20);

glVertex2d(x + 20, y + 40);

glVertex2d(x + 20, y + 60);

glVertex2d(x, y + 40);

glEnd();

glBegin(GL\_POLYGON);

glVertex2d(x, y + 60);

glVertex2d(x + 20, y + 80);

glVertex2d(x + 20, y + 100);

glVertex2d(x, y + 80);

glEnd();

// Draw the barrier legs

glColor3f(0.43, 0.16, 0.0);

glBegin(GL\_POLYGON);

glVertex2d(x - 20, y + 10);

glVertex2d(x - 5, y + 10);

glVertex2d(x, y + 15);

glVertex2d(x, y + 30);

glEnd();

glBegin(GL\_POLYGON);

glVertex2d(x + 40, y + 10);

glVertex2d(x + 25, y + 10);

glVertex2d(x + 20, y + 15);

glVertex2d(x + 20, y + 30);

glEnd();

glBegin(GL\_POLYGON);

glVertex2d(x + 40, y + 90);

glVertex2d(x + 25, y + 90);

glVertex2d(x + 20, y + 85);

glVertex2d(x + 20, y + 70);

glEnd();

glBegin(GL\_POLYGON);

glVertex2d(x - 20, y + 90);

glVertex2d(x - 5, y + 90);

glVertex2d(x, y + 85);

glVertex2d(x, y + 70);

glEnd();

}