Minigame: Ferește-te de fantome!

Autori: Buiciuc Ana-Maria
Răcășean Ivona-Maria

Personajul principal din proiectul nostru se numește Pătrățel și este foarte speriat de fantoma Casper care dorește să îl prindă. Tot ce trebuie să facem este să îl ajutăm să se ferească de ea de fiecare dată când îi apare în față. Atenție! Cu cât evităm fantoma mai mult, cu atât se enervează mai tare și își mărește viteza către Pătrățel. Sperăm să nu-l mănânce!

Pentru crearea personajelor am folosit primitive. După cum se intuiește și din nume, Pătrățel este format dintr-un pătrat mov, cu ochi și pupile pătrate, dar cu o gură dreptunghiulară și roșie, pentru a-l face să pară cât mai speriat. Fantoma Casper, în schimb, a fost desenată minuțios în GeoGebra după un contur. Gura acesteia este reprezentată printr-un cerc, iar ochii din două triunghiuri. Fundalul jocului este format, în principiu de cele 3 benzi pe care persoajele se mișcă și de marginile laterale bleumarin. Pentru a delimita benzile, am folosit linii discontinue, care de asemenea, fac jocul să pară mult mai dinamic.

Atunci când Pătrățel se mută de pe o bandă, pe alta, el își modifică unghiul de rotație cu orientarea spre direcția în care se deplasează: dacă se mută în sus, rotația se face spre stânga, iar dacă se mută în jos, rotația se face spre dreapta (compunerea de transformări constă în translație și rotație).

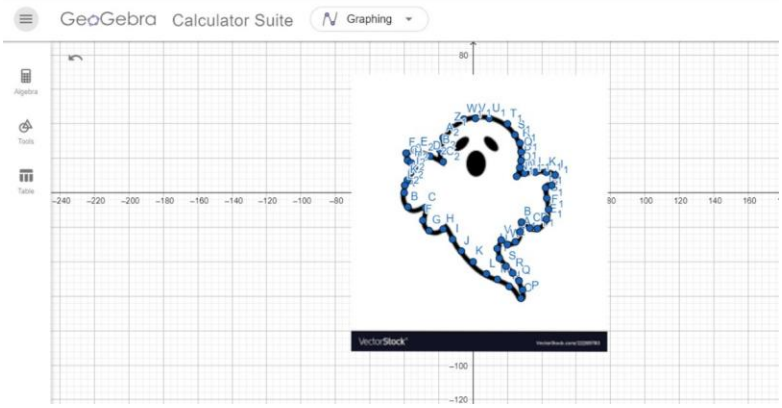
În momentul în care cele două personaje se intersectează, atât gura, cât și ochii Pătrățelului se modifică printr-o rotație, parând că se uită cruciș, iar ochii fantomei devin roșii și gura ovală.

Drept input interactiv am introdus prin meniu selectarea dificultății jocului (a vitezei cu care se deplasează fantoma): ușor, mediu sau greu, un meniu prin care poți alege culoarea Pătrățelului și a culoarelor de mers, dar și posibilitatea ieșirii din joc apăsând pe tasta Esc.

Noi considerăm că aspectele originale aduse proiectului sunt folosirea de rotații în momentul coliziunii personajelor, crearea fantomei în GeoGebra, dar și posibilitatea alegerii nivelului de dificultate prin meniu și costumizarea Pătrățelului.

În realizarea proiectului, amandouă am contribuit cu idei în aceeași măsură și am lucrat împreună pentru stabilirea fundalului și a personajelor. Ana este cea care a creat Pătrățelul, a venit cu ideea fantomei și a creat-o în GeoGebra, s-a ocupat de implemenarea meniurilor și de afișarea scorului actualizat în joc. Ivona s-a ocupat de rotația personajelor, de compunerea transformărilor și de liniile discontinue. Sursele folosite sunt cele din curs și laborator.

Captură GeoGebra – Crearea punctelor pentru conturul fantomei



Link către Demo:
https://drive.google.com/file/d/10Dg4gErutS54dLtdUxq4zJkOVfUZQib4/view?usp=sharing

Link GitHub:
anamariabuiciuc/MiniGame2D (github.com)

Cod sursă:

```
#include <iostream>
#include <string>
#include <math.h>
#include <windows.h>
#include <GL/freeglut.h>

using namespace std;

GLdouble left_m = -100.0;
GLdouble right_m = 700.0;
GLdouble bottom_m = -140.0;
GLdouble top_m = 460.0;
int myWindow = 0;
int myMenu = 1;
double ok = 1;
double j = 0.0;
double i = 0.0;
double contor = 0;
double loc_vert = 1000;
int vector[3] = { 0, 160, 320 };
double height = vector[rand() % 3];
int score = 0;
double timp;
```

```
int pct = 1000;
double rg, ros, rod = 0;
double r, g, b = 0;
double nr1 = 8;
double nr2 = 10;
double rotatie = 0.0;
int keybBackground, currentColor;
int x, y;

void init(void)
{
     glClearColor(0.0, 0.0, 0.15, 0.6);//culoare banda de mers
     glMatrixMode(GL_PROJECTION);
     glOrtho(left_m, right_m, bottom_m, top_m, -1.0, 1.0);
}

void RenderString(float x, float y, void* font, const unsigned
char* string)
{
     glColor3f(1.0f, 1.0f, 1.0f); //culoare font
     glRasterPos2f(x, y);
     glutBitmapString(font, string);
}

void startgame(void)
{

     if (height != j || (loc_vert > 90 || loc_vert < -90))
     {

          if (i < -380)
          {
               i = 0;
          }
          i = i - 2 * timp;

          loc_vert -= timp;

          if (loc_vert < -150)
          {
               score += 100;
               height = vector[rand() % 3];
               loc_vert = 1000;
          }
```

```
          if (score >= pct && pct <= 15000)
          {
               timp += 0.1;
               pct += 1000;

          }
          glutPostRedisplay();
     }
     else {

          ok = 0;
     }
}


void drawFantoma(void) {

     //corpul fantomei
     glColor3f(0.471, 0.667, 0.949);
     glBegin(GL_POLYGON);
     glVertex2f(-40, 0);
     glVertex2f(-37.8779504937335, -8.220157290478);
     glVertex2f(-27.8405080144665, -8.6764046758992);
     glVertex2f(-29.2092501707301, -15.9763628426389);
     glVertex2f(-25.5592710873603, -21.9075788531148);
     glVertex2f(-17.3468181497782, -20.9950840822724);
     glVertex2f(-11.8718495247235, -26.9263000927483);
     glVertex2f(-6.85312828509, -33.7700108740668);
     glVertex2f(0, -40);
     glVertex2f(7.7467880483893, -47.0011850512824);
     glVertex2f(14.1342514442864, -50.194916749231);
     glVertex2f(20.9779622256048, -54.301143218022);
     glVertex2f(27.8216730069232, -61.1448539993404);
     glVertex2f(28.8022800525159, -56.2626288275835);
     glVertex2f(26.6967479180559, -51.0542072318141);
     glVertex2f(23.0397710529412, -46.5106905206109);
     glVertex2f(19.1611592263044, -42.5212612132131);
     glVertex2f(15.2825473996676, -37.97774450201);
     glVertex2f(14.063555111296, -32.3260529831963);
     glVertex2f(16.3907222072781, -27.45008382971);
     glVertex2f(20, -30);
     glVertex2f(24.5912157835959, -28.4474411565595);
     glVertex2f(27.3616528026222, -22.5741146762238);
```

```
glVertex2f(28.1373751679496, -17.0332406381712);
glVertex2f(33.0133443214358, -20.4685825417638);
glVertex2f(37.4460435518779, -20.6902175032859);
glVertex2f(42.6544651476473, -15.2601609459944);
glVertex2f(43.8351469209556, -9.6160991463452);
glVertex2f(42.9062521582294, -3.1475761793096);
glVertex2f(42.1690295150011, 2.8976494951627);
glVertex2f(45.707698202497, 4.3720947816193);
glVertex2f(47.6922873643962, 10.3171597465896);
glVertex2f(42.5352973319763, 12.0017899845564);
glVertex2f(36.2498976050175, 12.0301025959391);
glVertex2f(30.0211231008242, 11.4072251455198);
glVertex2f(25.3495422226791, 9.5385927942618);
glVertex2f(27.1445616510744, 14.5825500616618);
glVertex2f(27.9241787038904, 18.6731495512845);
glVertex2f(27.9241787038904, 23.6781049901289);
glVertex2f(27.2338400226704, 28.8556450992783);
glVertex2f(24.2897747136441, 33.6220863940975);
glVertex2f(20, 40);
glVertex2f(9.4647894476066, 43.1825687409027);
glVertex2f(1.3965820534273, 43.1825687409027);
glVertex2f(-5.3269241083887, 42.7343349967817);
glVertex2f(-12.4986640143257, 37.8037638114499);
glVertex2f(-17.4292351996575, 31.976725137876);
glVertex2f(-19.2221701761417, 24.8049852319389);
glVertex2f(-17.4292351996575, 18.0814790701229);
glVertex2f(-25.0492088497156, 21.2191152789704);
glVertex2f(-32.2209487556527, 23.4602839995757);
glVertex2f(-35.3950684577688, 24.2414931671267);
glVertex2f(-37.0873199801594, 23.9312470546885);
glVertex2f(-38.9444549174687, 23.0120502554547);
glVertex2f(-38.7513673105101, 20.9416026984651);
glVertex2f(-38.0187800108894, 18.813707777508);
glVertex2f(-35.8633112492746, 17.3214601733131);
glVertex2f(-34.0394530663697, 14.8343808329883);
glVertex2f(-36.1949218279846, 11.8498856245985);
glVertex2f(-38.1845853002444, 7.3731428120139);
glVertex2f(-39.6768329044393, 4.3886476036241);
glEnd();

float theta;
//ochi fantoma
//stang
glColor3f(0, 0, 0);
```

```
     glColor3f(r, g, b);
     glBegin(GL_TRIANGLES);
     glVertex2f(-5, 10);
     glVertex2f(-20, 15);
     glVertex2f(-10, 33);
     glEnd();

     //drept
     glColor3f(0, 0, 0);
     glColor3f(r, g, b);
     glBegin(GL_TRIANGLES);
     glVertex2f(5, 10);
     glVertex2f(20, 15);
     glVertex2f(10, 33);
     glEnd();

     //gura
     glColor3f(0, 0, 0);
     glBegin(GL_POLYGON);
     for (int i = 0; i < 360; i++) {
          theta = i * 3.142 / 180;
          glVertex2f(nr1 * cos(theta), -5 + nr2 * sin(theta));
     }
     glEnd();

}


void drawScene(void)
{
     glClear(GL_COLOR_BUFFER_BIT);

     glColor3f(0.00, 0.10, 0.30);

     if (myMenu == 1)
     {
          RenderString(200.0f, 425.0f,
GLUT_BITMAP_TIMES_ROMAN_24, (const unsigned char*)"Fereste-te de
fantome!");
          RenderString(200.0f, 380.0f,
GLUT_BITMAP_TIMES_ROMAN_24, (const unsigned char*)"Alege
dificultatea.");

          glColor3f(0.0f, 0.8f, 0.1f);
```

```
          glRecti(250, 250, 400, 300);
          RenderString(300.0f, 268.0f,
GLUT_BITMAP_TIMES_ROMAN_24, (const unsigned char*)"Usor");

          glColor3f(1.0f, 0.5f, 0.0f);
          glRecti(250, 150, 400, 200);
          RenderString(295.0f, 168.0f,
GLUT_BITMAP_TIMES_ROMAN_24, (const unsigned char*)"Mediu");

          glColor3f(1.0f, 0.0f, 0.0f);
          glRecti(250, 50, 400, 100);
          RenderString(300.0f, 68.0f,
GLUT_BITMAP_TIMES_ROMAN_24, (const unsigned char*)"Greu");

          drawFantoma();

     }

     else {
          // Margine 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();

          //Margine 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(200.0f, 425.0f,
GLUT_BITMAP_TIMES_ROMAN_24, (const unsigned char*)"Fereste-te de
fantome!");

          //Afisare scor pe display
          glColor3f(1.0, 1.0, 1.0);
          glRasterPos2f(225.0f, -120.0f);
          string stringScore = "SCOR: " + to_string(score);
```

```
            glutBitmapString(GLUT_BITMAP_TIMES_ROMAN_24, (const
unsigned char*)stringScore.c_str());

            // Delimitare sosea
            glLineWidth(6);
            glColor3f(0, 0.17, 0.5);

            // Delimitam soseaua de marginea de jos
            glBegin(GL_LINES);
            glVertex2i(-100, -80);
            glVertex2i(1500, -80);
            glEnd();

            // Delimitam soseaua de marginea de sus
            glBegin(GL_LINES);
            glVertex2i(-100, 400);
            glVertex2i(1500, 400);
            glEnd();

            // Liniile intrerupte
            glPushMatrix();
            glTranslated(i, 0.0, 0.0);
            glLineStipple(8, 0x00FF);
            glEnable(GL_LINE_STIPPLE);

            glBegin(GL_LINES);
            glVertex2i(-100, 80);
            glVertex2i(1500, 80);
            glEnd();

            glBegin(GL_LINES);
            glVertex2i(-100, 240);
            glVertex2i(1500, 240);
            glEnd();
            glPopMatrix();

            glDisable(GL_LINE_STIPPLE);

            //desenam caracterul Patratel
            glPushMatrix();
            glTranslated(0.0, j, 0.0);

            glPushMatrix();
```

```
          glRotated(rotatie, 0.0, 0.0, 1.0); //rotire cand
schimba banda de mers

          if (currentColor == 0)
               glColor3f(0.5, 0.3, 0.5); //mov
          if (currentColor == 1)
               glColor3f(0.96, 0.55, 0.87); // roz
          glRecti(-45, -45, 45, 45);//corpul in forma de patrat

          if (ok == 0) //atunci cand e lovit
          {
               ros = 8; //rotatie ochi stang
               rod = -8; //rotatie ochi drept
               rg = 8;

               //schimba culoarea ochilor fantomei cand playerul
este lovit
               r = 1;
               g = 0.1;
               b = 0.1;

               //schimba forma gurii fantomei
               nr1 = 12;
               nr2 = 8;
          }

          //ochi drept
          glPushMatrix();
          glRotated(rod, 0.0, 0.0, 1.0);
          glColor3f(1, 1, 1);
          glRecti(10, 10, 30, 30);
          glPopMatrix();

          //pupila dreapta
          glColor3f(0, 0, 0);
          glRecti(15, 15, 25, 25);

          //ochi stang
          glPushMatrix();
          glRotated(ros, 0.0, 0.0, 1.0);
          glColor3f(1, 1, 1);
          glRecti(-30, 10, -10, 30);
          glPopMatrix();
```

```
          //pupila stanga
          glColor3f(0, 0, 0);
          glRecti(-25, 15, -15, 25);

          //gura
          glPushMatrix();
          glRotated(rg, 0.0, 0.0, 1.0);
          glColor3f(1, 0, 0.5);
          glRecti(-15, -10, 15, -20);
          glPopMatrix();

          glPopMatrix();
          glPopMatrix();

          //Game Over
          if (ok == 0) {
               RenderString(250.0f, 200.0f, GLUT_BITMAP_8_BY_13,
(const unsigned char*)"AI FOST MANCAT...");
          }

          if (contor == 1 && (j != 160 && j != 320))
               j = j + 1;
          else if (contor == -1 && (j != 160 && j != 0))
               j = j - 1;
          else {
               contor = 0;
               rotatie = 0;

          }

          //desenam fantoma (adversara)
          glPushMatrix();
          glTranslated(loc_vert, height, 0.0);

          drawFantoma();

          glPopMatrix();
     }

     startgame();
     glutPostRedisplay();
     glutSwapBuffers();
     glFlush();
```

```
     myWindow = glutGetWindow();
}

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)
     {
          if (j < 320)
          {
               contor = 1;
               j += 1;
               rotatie = 15;
          }
          glutPostRedisplay();
     }
}

void miscajos(void)
{
     if (ok != 0)
     {
          if (j > 0)
          {
               contor = -1;
               j -= 1;
               rotatie = -15;

          }
          glutPostRedisplay();
     }
}

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

```
	switch (key) {
	case GLUT_KEY_UP:
		miscasus();
		break;
	case GLUT_KEY_DOWN:
		miscajos();
		break;
	}

}

void processNormalKeys(unsigned char key, int x, int y) {

	if (key == 27) //iesire din joc cu tasta ESCAPE
	{
		glutDestroyWindow(myWindow);
		exit(0);
	}
}

void Initialize(int key)
{
	switch (key)
	{
	case 0:
		glClearColor(0.0, 0.0, 0.15, 0.6);//bleumarin
		keybBackground = 0;
		break;
	case 1:
		glClearColor(0.47, 0.47, 0.47, 0.0); //gri
		keybBackground = 1;
		break;
	case 2:
		glClearColor(0.1, 0.5, 0.19, 0.0);//verde inchis
		keybBackground = 2;
		break;
	}
}

void callback_Main(int key)
{
	if (key == 0)
	{
```

```
          exit(0);
     }
}

void callback_Color(int key)
{
     currentColor = key;
}

//meniul de start - alegere dificultate joc
void mouse(int button, int state, int mx, int my)
{
     if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
     {
          x = mx;
          y = 600 - my;//schimbam in functie de ecran
          if (myMenu == 1 && mx >= 350 && mx <= 500 && my >= 160
&& my <= 210) {
               timp = 0.15;
               glutDisplayFunc(drawScene);
               myMenu = 0;
          }

          if (myMenu == 1 && mx >= 350 && mx <= 500 && my >= 260
&& my <= 310) {
               timp = 0.3;
               glutDisplayFunc(drawScene);
               myMenu = 0;
          }

          if (myMenu == 1 && mx >= 350 && mx <= 500 && my >= 360
&& my <= 410) {
               timp = 0.5;
               glutDisplayFunc(drawScene);
               myMenu = 0;
          }
     }
}


int main(int argc, char** argv)
{
     int menuMain, menuBackground, menuColor;
     glutInit(&argc, argv);
```

```
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(800, 600);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Ai grija sa nu te prinda fantomele!");
    init();
    glutDisplayFunc(drawScene);
    glutReshapeFunc(reshape);
    glutMouseFunc(mouse);
    glutSpecialFunc(keyboard);
    glutKeyboardFunc(processNormalKeys);

    menuBackground = glutCreateMenu(Initialize);
    glutAddMenuEntry("Bleumarin", 0);
    glutAddMenuEntry("Gri", 1);
    glutAddMenuEntry("Verde", 2);

    menuColor = glutCreateMenu(callback_Color);
    glutAddMenuEntry("Mov ", 0);
    glutAddMenuEntry("Roz ", 1);

    menuMain = glutCreateMenu(callback_Main);

    glutAddSubMenu("Culoare sosea", menuBackground);
    glutAddSubMenu("Culoare patratel ", menuColor);
    glutAddMenuEntry("Iesire ", 0);
    glutAttachMenu(GLUT_RIGHT_BUTTON);

    glutMainLoop();
}
```