**ПОЯСНЕНИЕ КОДА**

**Практическая работа №7. Графическая мини-игра "Битва с монстром"**

**Описание программы:**

Это 2D-экшен игра, в которой игрок управляет рыцарем и сражается с монстрами. Цель — как можно дольше выживать, побеждать монстров и зарабатывать очки. По ходу игры можно собирать бонусы, чтобы восстанавливать здоровье. После победы над монстром появляется следующий, более сильный. Если здоровье игрока опустится до 0 — игра заканчивается.

РАЗБОР КОДА

#include "raylib.h"

#include <cmath>

#include <cstdlib>

#include <ctime>

const int screenWidth = 1500;

const int screenHeight = 750;

**РАЗМЕР ИГРОВОГО ОКНА**

enum GameState { MENU, GAME, PAUSE, GAME\_OVER };

**ЭТО СОСТОЯНИЕ ИГРЫ. МОЖЕТ БЫТЬ МЕНЮ , В САМОЙ ИГРЕ , В ПАУЗЕ ИЛИ В ЭКРАНЕ ПРОИГРЫША**

struct Player {

Vector2 position;

int health;

float speed;

Texture2D texture;

float attackCooldown;

float scale;

float damageTimer;

};

**СТРУКТУРА ИГРОКА. (Позиция, здоровье, скорость, текстура, время до следующей атаке и тд)**

struct Monster {

Vector2 position;

float speed;

Texture2D texture;

float scale;

int health;

float respawnTimer;

};

**СТРУКТУРА МОНСТРА ( позиция, скорость, текстура, и тд)**

struct Bonus {

Vector2 position;

Texture2D texture;

bool active;

float timer;

float scale;

};

**СТРУКТУРА БОНУСА ( позиция, текстура, активность, и тд)**

int main()

{

InitWindow(screenWidth, screenHeight, "Fight with the monster - Sagantay Adil CS-104(s)");

SetTargetFPS(60);

srand(time(0));

**ОКНО С НАЗВАНИЕМ, УСТАНОВЛЕНИЕ FPS, ГЕНЕРАЦИЯ СЛУЧАЙНЫХ ЧИСЕЛ НА ОСНОВЕ ТЕКУЩЕГО ВРЕМЕНИ**

Texture2D itemTexture = LoadTexture("forest.png");

Texture2D knightTexture = LoadTexture("knight.png");

Texture2D monsterTexture1 = LoadTexture("monster.png");

Texture2D monsterTexture2 = LoadTexture("monster2.png");

Texture2D monsterTexture3 = LoadTexture("monster3.png");

Texture2D bonusTexture = LoadTexture("bonus.png");

**ЗАГРУЖАЮТСЯ ВСЕ КАРТИНКИ ДЛЯ ИГР:**

GameState currentState = MENU;

Player player = { {screenWidth / 3, screenHeight / 2}, 100, 200.0f, knightTexture, 0.0f, 0.2f, 0.0f };

Bonus bonus = { { 0, 0 }, bonusTexture, false, 0.0f, 0.2f };

**НАЧИНАЕТСЯ С МЕНЮ, СОЗДАЕТСЯ ИГРОКА, БОНУС**

int currentMonsterIndex = 0;

Monster monsters[3] = {

{ {screenWidth / 2, screenHeight / 2}, 100.0f, monsterTexture1, 0.2f, 50, 0.0f },

{ {screenWidth / 2, screenHeight / 2}, 120.0f, monsterTexture2, 0.2f, 60, 0.0f },

{ {screenWidth / 2, screenHeight / 2}, 80.0f, monsterTexture3, 0.19f, 70, 0.0f }

};

**СОЗДАЕТСЯ 3 МОНСТРА**

Monster\* monster = &monsters[currentMonsterIndex];

**УКАЗАТЕЛЬ НА ТЕКУЩЕГО МОНСТРА, КОТОРЫЙ ИДЕТ В БОЙ**

int score = 0;

float gameTime = 0.0f;

float hitFlashTimer = 0.0f;

bool playerDamaged = false;

const int initialMonsterHealth[3] = { 50, 70, 100 };

while (!WindowShouldClose())

{

float deltaTime = GetFrameTime();

if (IsKeyPressed(KEY\_ENTER)) currentState = GAME;

if (IsKeyPressed(KEY\_P)) currentState = (currentState == PAUSE) ? GAME : PAUSE;

if (IsKeyPressed(KEY\_M)) {

currentState = MENU;

player.health = 100;

player.position = { screenWidth / 3, screenHeight / 2 };

score = 0;

gameTime = 0.0f;

hitFlashTimer = 0.0f;

playerDamaged = false;

currentMonsterIndex = 0;

monster = &monsters[currentMonsterIndex];

monster->health = initialMonsterHealth[currentMonsterIndex];

monster->position = { (float)(rand() % screenWidth), (float)(rand() % screenHeight) };

monster->respawnTimer = 0.0f;

}

if (IsKeyPressed(KEY\_Q)) currentState = GAME\_OVER;

**ОБРАБОТКА КЛАВИШИ**

**ENTER – НАЧАЛО ИГРЫ**

**P - ПАУЗА**

**M – ВЫХОД НА ГЛАВНОЕ МЕНЮ, СБРАСЫВАЯ ПАРАМЕТРОВ ИГРЫ**

**Q – ПРОИГРЫШ**

if (currentState == GAME)

{

gameTime += deltaTime;

// Движение игрока

if (IsKeyDown(KEY\_RIGHT)) player.position.x += player.speed \* deltaTime;

if (IsKeyDown(KEY\_LEFT)) player.position.x -= player.speed \* deltaTime;

if (IsKeyDown(KEY\_UP)) player.position.y -= player.speed \* deltaTime;

if (IsKeyDown(KEY\_DOWN)) player.position.y += player.speed \* deltaTime;

if (hitFlashTimer > 0.0f) hitFlashTimer -= deltaTime;

**ДВИЖЕНИЕ ИГРОКА ПО СТРЕКАМ**

// Логика бонусов

bonus.timer -= deltaTime;

if (!bonus.active && bonus.timer <= 0.0f) {

bonus.position = { (float)(rand() % (screenWidth - bonus.texture.width)), (float)(rand() % (screenHeight - bonus.texture.height)) };

bonus.active = true;

bonus.timer = 10.0f;

}

**ЕСЛИ БОНУС НЕ АКТИВЕН ОН ЧЕРЕЗ 10 СЕКУНД ПОЯВЛЯЕТСЯ ПОСЛЕ НАЧАЛА ИГРЫ**

// Ограничение движения игрока по экрану

float knightWidth = knightTexture.width \* player.scale;

float knightHeight = knightTexture.height \* player.scale;

if (player.position.x < 0) player.position.x = 0;

if (player.position.x > screenWidth - knightWidth) player.position.x = screenWidth - knightWidth;

if (player.position.y < 0) player.position.y = 0;

if (player.position.y > screenHeight - knightHeight) player.position.y = screenHeight - knightHeight;

**ОГРАНИЧЕНИЕ ДВИЖЕНИЯ ИГРОКА ПО ЭКРАНУ, ЧТОБЫ НЕ МОГ ВЫЙТИ ЗА ЕГО ГРАНИЦУ**

// Логика монстра

if (monster->health > 0) {

Vector2 direction = { player.position.x - monster->position.x, player.position.y - monster->position.y };

float length = sqrt(direction.x \* direction.x + direction.y \* direction.y);

if (length != 0) {

direction.x /= length;

direction.y /= length;

}

monster->position.x += direction.x \* monster->speed \* deltaTime;

monster->position.y += direction.y \* monster->speed \* deltaTime;

}

else {

monster->respawnTimer -= deltaTime;

if (monster->respawnTimer <= 0) {

currentMonsterIndex = (currentMonsterIndex + 1) % 3;

monster = &monsters[currentMonsterIndex];

monster->position = { (float)(rand() % screenWidth), (float)(rand() % screenHeight) };

monster->health = initialMonsterHealth[currentMonsterIndex];

monster->respawnTimer = 0.0f;

}

}

**ЛОГИКА МОНСТРОВ**

**ЕСЛИ МОНСТР ЖИВ – ОН ПРЕСЛЕДУЕТ ЗА ИГРОКОМ**

**ЕСЛИ МОНСТР ПОБЕЖДЕН – ПОЯВЛЯЕТСЯ ЧЕРЕЗ 3 СЕКУНДЫ СЛЕДУЮЩИЙ МОНСТР**

// Проверка на столкновение игрока и монстра

Rectangle playerRect = { player.position.x, player.position.y, knightWidth, knightHeight };

Rectangle monsterRect = { monster->position.x, monster->position.y, monster->texture.width \* monster->scale, monster->texture.height \* monster->scale };

if (monster->health > 0 && CheckCollisionRecs(playerRect, monsterRect)) {

if (player.damageTimer <= 0.0f) {

player.health -= 5;

player.damageTimer = 1.5f;

playerDamaged = true;

}

}

if (player.damageTimer > 0.0f) {

player.damageTimer -= deltaTime;

if (player.damageTimer <= 1.2f) playerDamaged = false;

}

**СТОЛКНОВЕНИЕ ИГРОКА С МОНСТРОМ**

**ЕСЛИ ЕСТЬ СОПРИКОСНОВЕНИЕ – ИГРОК ПОЛУЧАЕТ УРОН**

// Атака игрока

if (IsKeyPressed(KEY\_SPACE) && player.attackCooldown <= 0.0f) {

if (monster->health > 0 && CheckCollisionRecs(playerRect, monsterRect)) {

monster->health -= 25;

hitFlashTimer = 0.2f;

if (monster->health <= 0) {

monster->respawnTimer = 3.0f;

score += 10;

}

}

player.attackCooldown = 1.0f;

}

if (player.attackCooldown > 0.0f) player.attackCooldown -= deltaTime;

**УДАР ПО МОНСТРУ. ЕСТЬ ЗАДЕРЖКА МЕЖДУ УДАРАМИ attackCooldown**

// Проверка бонуса

if (bonus.active) {

Rectangle bonusRect = { bonus.position.x, bonus.position.y, bonus.texture.width \* bonus.scale, bonus.texture.height \* bonus.scale };

if (CheckCollisionRecs(playerRect, bonusRect)) {

player.health += 20;

if (player.health > 100) player.health = 100;

bonus.active = false;

bonus.timer = 10.0f;

}

}

**ПРИ СОПРИКОСНОВЕНИИ БОНУСА – ИГРОК ПОЛУЧАЕТ 20HP ( МАКСИМУМ 100 HP)**

// Проверка на "Game Over"

if (player.health <= 0) {

currentState = GAME\_OVER;

}

}

**ПРОВЕРКА НА ПРОИГРЫШ**

**ЕСЛИ У ИГРОКА 0 HP – ИГРА ЗАКАНЧИВАЕТСЯ**

BeginDrawing();

ClearBackground(RAYWHITE);

**НАЧАЛО РИСОВАНИЯ**

switch (currentState) {

case MENU:

DrawText("Game Menu. Press ENTER to start", screenWidth / 3, screenHeight / 3, 20, DARKGRAY);

DrawText("Press Q to simulate Game Over", screenWidth / 3, screenHeight / 3 + 30, 20, DARKGRAY);

break;

case GAME:

DrawTexturePro(itemTexture, { 0.0f, 0.0f, (float)itemTexture.width, (float)itemTexture.height }, { 0, 0, (float)screenWidth, (float)screenHeight }, { 0.0f, 0.0f }, 0.0f, WHITE);

DrawText("Game in progress. Press P to pause", screenWidth / 3, 20, 20, BLACK);

**ЭТО ВЫБОР ТОГО В КАКОМ СОСТОЯНИИ НАХОДИТСЯ В ИГРЕ: МЕНЮ, ПАУЗА, ИГРА**

**РИСУЕТ СОСТОЯНИЕ ИГРА**

Color playerColor = playerDamaged ? RED : WHITE;

DrawTextureEx(player.texture, player.position, 0.0f, player.scale, playerColor);

**ЕСЛИ ИГРОК РАНЕН – ВСПЫШКА КРАСНОГО ЦВЕТА**

if (monster->health > 0) {

Color monsterColor = (hitFlashTimer > 0.0f) ? RED : WHITE;

DrawTextureEx(monster->texture, monster->position, 0.0f, monster->scale, monsterColor);

DrawText(TextFormat("%d", monster->health), monster->position.x, monster->position.y - 20, 20, RED);

}

if (bonus.active) DrawTextureEx(bonus.texture, bonus.position, 0.0f, 0.7f, WHITE);

**ПОКАЗЫВАЕТ СКОЛЬКО ЗДОРОВЬЕ МОНСТРА НАД ГОЛОВОЙ МОНСТРА**

**ЕСЛИ ИГРОК УДАРИТ МОНСТР – ВСПЫШКА КРАСНОГО ЦВЕТА**

**ЕСЛИ БОНУС АКТИВЕН – ОН РИСУЕТ**

DrawRectangle(10, 10, 250, 80, Fade(GRAY, 0.5f));

DrawRectangleLines(10, 10, 250, 80, BLACK);

DrawText(TextFormat("HP: %d", player.health), 20, 20, 20, RED);

DrawText(TextFormat("Time: %.1f s", gameTime), 20, 40, 20, BLACK);

DrawText(TextFormat("Score: %d", score), 20, 60, 20, DARKBLUE);

break;

**РИСУЕТ СОСТОЯНИЕ ИГРЫ, РАМКА, СЧЕТ, ЗДОРОВЬЯ, ВРЕМЯ**

case PAUSE:

DrawText("Paused. Press P to continue", screenWidth / 3, screenHeight / 3, 20, DARKGRAY);

DrawText("Press M to return to menu", screenWidth / 3, screenHeight / 3 + 30, 20, DARKGRAY);

break;

case GAME\_OVER:

DrawText("Game Over. Press M to return to menu", screenWidth / 3, screenHeight / 3, 20, RED);

break;

}

**РИСУЕТ В ПАУЗЕ СООТВЕТСВУЮЩИЙ ТЕКСТ**

if (playerDamaged) DrawRectangle(0, 0, screenWidth, screenHeight, Fade(RED, 0.2f));

ЕСЛИ ИГРОК ПОЛУЧИЛ УДАР ОТ МОНСТРА – ВСПИХЫВАЕТ КРАСНЫМ ЦВЕТОМ

EndDrawing();

}

UnloadTexture(bonusTexture);

UnloadTexture(itemTexture);

UnloadTexture(knightTexture);

UnloadTexture(monsterTexture1);

UnloadTexture(monsterTexture2);

UnloadTexture(monsterTexture3);

CloseWindow();

return 0;

**освобождают ресурсы, удаляя текстуры из памяти.**