

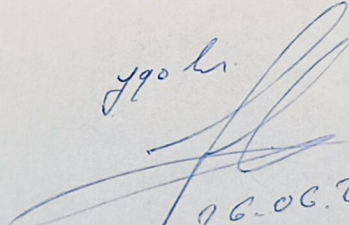
Министерство образования Российской Федерации
Пензенский государственный университет
Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе
по курсу «Программирование»
на тему «Арканойд»

Выполнил:
студент группы 23ВВВ4
Королёв Д.В.

Принял:
к.т.н., доцент Голотенков Н.О.


26.06.24

Пенза, 2024

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет Вычислительной техники
Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ



М.А. Митрохин

« ____ » _____ 2024г

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Королёву Дмитрию Вячеславовичу Группа 23BVB4

Тема проекта: *Разработка программы сложной структуры методом нисходящего программирования.*

Исходные данные (технические требования) на проектирование

Разработка компьютерной игры «Арканойд». Программа должна быть выполнена в текстовом или графическом интерфейсе. Игровое поле должно быть размером не менее 70% от экрана монитора. Управление – клавиатура. Список последних 10 рекордсменов должен сохраняться в файл и выводиться на экран по запросу.

Обязательные требования к программе:

1. Многомодульность.
2. Использование сложных типов данных – структур, списков, файлов.
3. Режим работы видеосистемы – текстовый / графический.
4. Устройство ввода информации – клавиатура, мышь.
5. Пользовательский интерфейс должен быть построен на основе меню и панели инструментов.
6. Наличие заставки.
7. Операционная система MS Windows.
8. Язык программирования – Си и Ассемблер.
9. Среда разработки ПО – Microsoft Visual Studio.

Объем работы по курсу

1. Расчетная часть

Разработка программы.

2. Графическая часть

Схема данных, схема ресурсов системы, схема работы системы, иерархическая структура программы, схема взаимодействия программы.

3. Экспериментальная часть

Отладка программы.

Срок выполнения проекта по разделам

В соответствии с графиком.

Дата выдачи задания "9" февраля 2024г.

Дата защиты проекта "26" 06. 24

Руководитель Голотенков Н.О.

Задание получил "9" февраля 2024г.

Студент Королёв Д.В.

Оглавление

Введение	3
1. Постановка задачи	4
2. Выбор решения	5
3. Описание разработки программы	5
4. Описание программы.....	6
5. Отладка и тестирование.....	22
6. Руководство пользователя	23
7. Заключение.....	24
8. Список используемых источников.....	25
Приложение А Листинги программы	26
Приложение В Результаты работы программы	33
Приложение С Результаты работы программы	34

Введение

Arkanoid - это классическая аркадная игра, разработанная компанией Taito в 1986 году. Она является одним из самых популярных представителей жанра "breakout", где игрок управляет платформой, отскакивая мяч от нее, чтобы разрушать блоки на экране. Игра Arkanoid быстро стала популярной благодаря своей простоте и захватывающему геймплею.

Разработка консольной игры Arkanoid на языке Си представляет собой увлекательное и познавательное занятие, которое позволяет развить навыки программирования и логического мышления. Создание своей версии Arkanoid на Си позволит погрузиться в процесс создания игровой логики, управления игровыми объектами и обработки пользовательского ввода. Игра Arkanoid не только предоставляет развлечение и возможность провести время с пользой, но также способствует развитию рефлексов, координации движений и принятию быстрых решений.

Разработка консольной игры Arkanoid на языке Си может быть интересным и полезным проектом для тех, кто хочет погрузиться в мир разработки игр и усовершенствовать свои навыки программирования. Кроме того, процесс разработки Arkanoid на языке Си предоставляет отличную возможность изучить такие концепции, как физика столкновений, алгоритмы определения столкновений и визуализация графики. Понимание этих аспектов необходимо для создания реалистичного и отзывчивого игрового опыта.

В процессе разработки игры Arkanoid на Си, программисты сталкиваются с задачами, связанными с оптимизацией производительности и управлением памятью, что является ключевым навыком для любого разработчика программного обеспечения. Также, это хорошая практика для работы с пользовательским интерфейсом и взаимодействием с пользователем, поскольку игра должна быть интуитивно понятной и легкой в освоении. Разработка Arkanoid на Си также может включать создание

уровней разной сложности, добавление специальных эффектов и бонусов, которые делают игру более интересной и динамичной. Это требует творческого подхода и способности мыслить нестандартно, что полезно для развития креативности. В конечном итоге, создание Arkanoid на Си не только улучшает технические навыки, но и предоставляет полную свободу для экспериментов и инноваций, позволяя разработчикам воплощать свои идеи в жизнь и видеть, как их творения оживают на экране. Это может быть чрезвычайно удовлетворительным опытом, который вдохновляет на новые проекты и достижения в области программирования и разработки игр.

1. Постановка задачи

При разработке игры Arkanoid необходимо поставить перед собой ряд задач, которые позволят создать качественный и интересный игровой продукт. Одной из ключевых задач является многомодульная структура проекта. Использование модульного подхода позволит разделить код на логически связанные части, что упростит его поддержку и дальнейшее расширение. К примеру, можно выделить модули для обработки игровой логики, управления графикой, взаимодействия с пользователем и другие. Для работы с игровыми объектами, такими как платформа, мяч, блоки и т.д., необходимо использовать сложные типы данных, такие как структуры. С их помощью можно описать характеристики и поведение каждого объекта, что обеспечит более гибкое управление ими в игре.

Устройство ввода информации в игре Arkanoid осуществляется с помощью клавиатуры. Необходимо предусмотреть обработку нажатий клавиш для управления платформой и другими игровыми элементами. Важным аспектом является создание пользовательского интерфейса. Также стоит уделить внимание наличию заставки. Для разработки игры Arkanoid на операционной системе Windows с использованием языков программирования Си и ассемблера рекомендуется использовать среду разработки Visual Studio.

Это интегрированная среда разработки, которая обладает широкими возможностями для создания игр и обеспечивает удобный рабочий процесс разработки и отладки игрового приложения

2. Выбор решения

При запуске программы открывается приветственный экран. Далее появляется меню, состоящее из двух пунктов:

1. Начать игру: после появляется окно для ввода имени пользователя и начинается игра. У игрока есть 3 жизни. Если 3 жизни кончились или после того, как пользователь нажимает “escape”, игра завершается и в файл “high_scores.txt” записывается его имя и максимальное количество очков, которое набрал пользователь.

2. Показать рекорды: появляется список, состоящий из 10 рекордсменов, которые получили максимальный результат

Навигация в меню осуществляется с помощью клавиатуры, вводом цифр 0 или 1.

3. Описание разработки программы

Разработанная программа состоит из нескольких модулей, которые реализовывались в следующем порядке:

- 1) main.cpp - основной файл программы, содержит всю логику, меню и т.д.
- 2) ball.cpp - содержит функции, отвечающие за инициализацию мячика, его движения и помещения в поле с игрой

По порядку реализации модулей можно понять, что использовался нисходящий подход реализации программы. Этот метод основан на том, что сначала проектируются основные компоненты программы, а затем уже дорабатываются ее мелкие детали. Это позволяет упростить ее тестирование,

так как каждый модуль тестируется сразу же после его реализации, что помогает в создании более структурированных программ

4. Описание программы

Эта программа представляет собой консольную версию классической игры "Арканоид", написанную на языке С. Вот её описание:

Директивы препроцессора: Программа начинается с нескольких директив препроцессора для подключения необходимых заголовочных файлов и определения констант для размеров игры и количества блоков.

Структуры данных: Определяются структуры для блоков (Tblock), ракетки (Tracket) и рекордов (HighScore).

Глобальные переменные: Глобально объявляются массивы и переменные для хранения состояния игры, такие как игровое поле (mas), ракетка, мяч, блоки и счётчики очков.

Функции игры:

autoMoveBall(): Управляет движением мяча и логикой столкновений, включая столкновения со стенами и блоками, и увеличивает скорость мяча после каждых 10 ударов.

initRacket(), putRacket(): Инициализируют и отображают ракетку игрока.

initBlocks(), putBlocks(): Инициализируют и отображают блоки, которые может поразить мяч.

init(): Инициализирует игровую область с границами и пустым пространством.

show(): Отрисовывает игровую область и отображает текущий счёт и рекорд.

moveRacket(int x): Перемещает ракетку на основе ввода игрока.

setCur(int x, int y): Устанавливает позицию курсора в окне консоли для рисования.

`greeting()`: Отображает приветственное сообщение с ASCII-артом.

Функции рекордов:

`compareScores()`: Функция сравнения для сортировки рекордов.

`displayHighScores()`: Читает рекорды из файла и отображает их.

`updateHighScores()`: Обновляет файл рекордов новыми записями.

Основной игровой цикл (`main()`):

Отображает приветствие и меню для начала игры или просмотра рекордов.

Если игра начинается, запрашивает никнейм и инициализирует состояние игры.

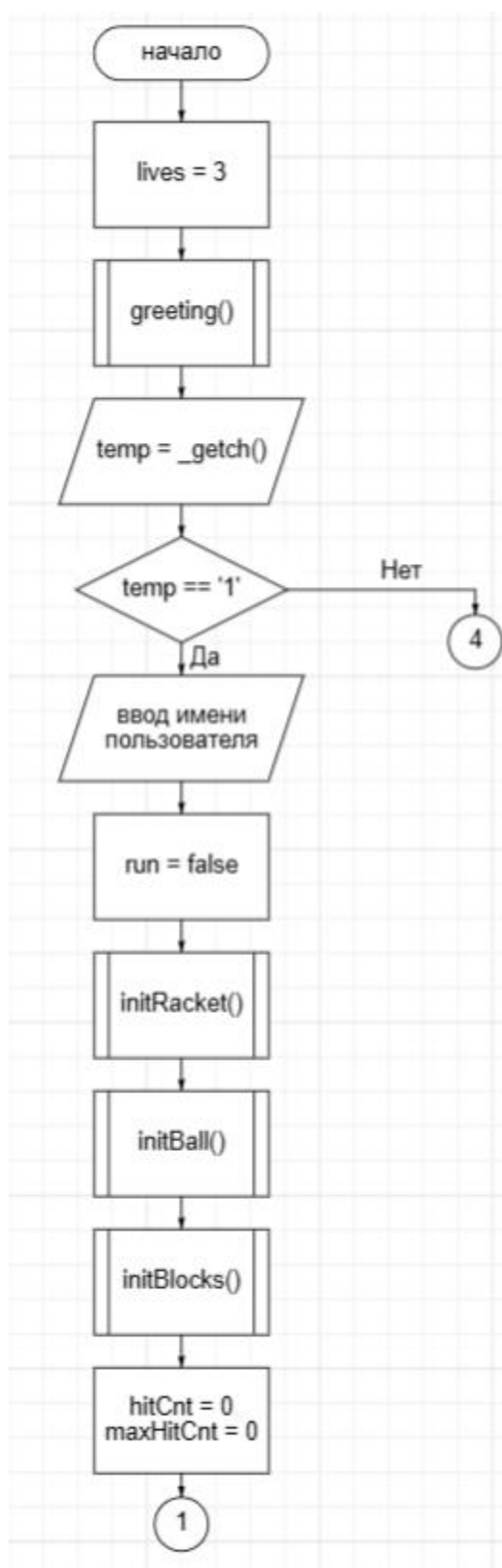
Игровой цикл обрабатывает движение мяча, управление ракеткой и логику окончания игры.

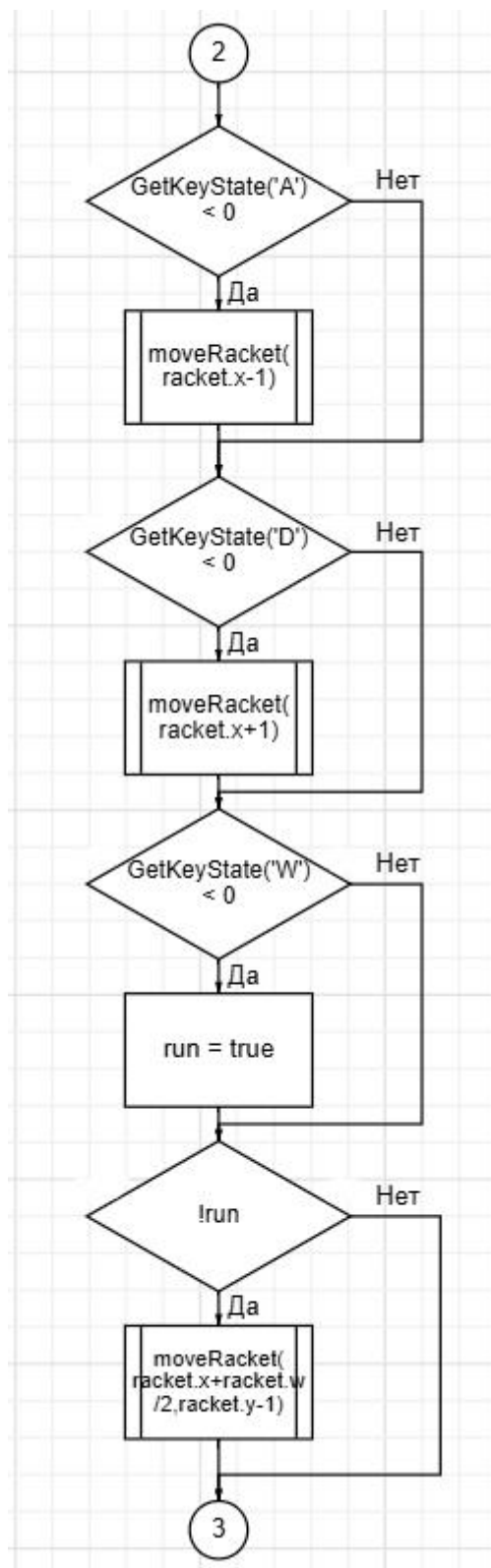
Ввод игрока обрабатывается для перемещения ракетки и запуска движения мяча.

Игра обновляет и отображает счёт, и обновляет рекорды после окончания игры.

Игра управляется с помощью клавиш 'A' и 'D' для перемещения ракетки и 'W' для начала движения мяча. Игра продолжается до тех пор, пока игрок не потеряет все жизни или не выйдет с помощью клавиши `Escape`. Рекорды записываются и отображаются, что стимулирует повторную игру. Использование консольных функций, таких как `SetConsoleCursorPosition` и `GetKeyState`, указывает на то, что эта игра предназначена для запуска на платформах Windows. Код хорошо организован с чёткими функциональными обязанностями, что делает его хорошим примером простого, но полного игрового проекта на C. В целом, это текстовое представление Арканоида, демонстрирующее основные концепции программирования, такие как структуры, файловый ввод-вывод и дизайн игрового цикла.

Схема программы
Схема основной программы





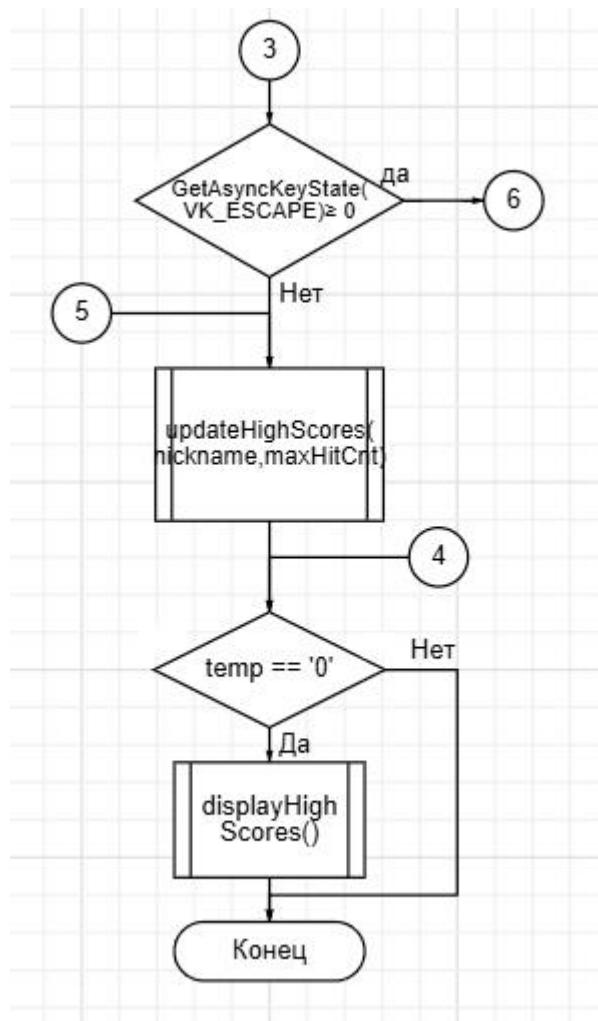


Схема функции initRacket

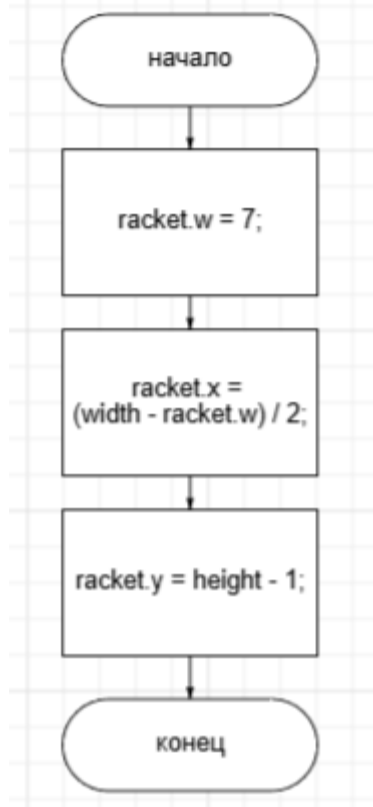


Схема функции initBall()

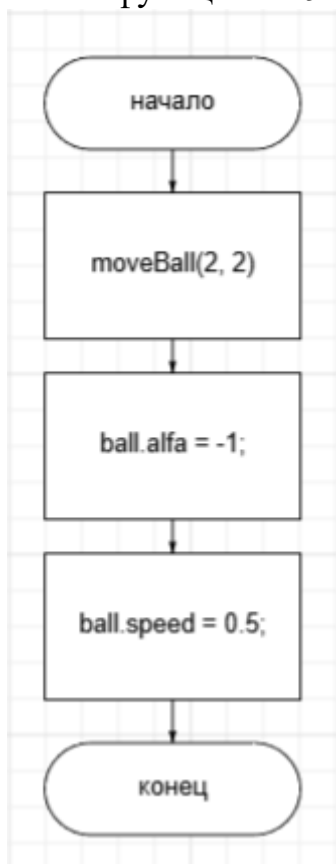


Схема функции initBlocks()

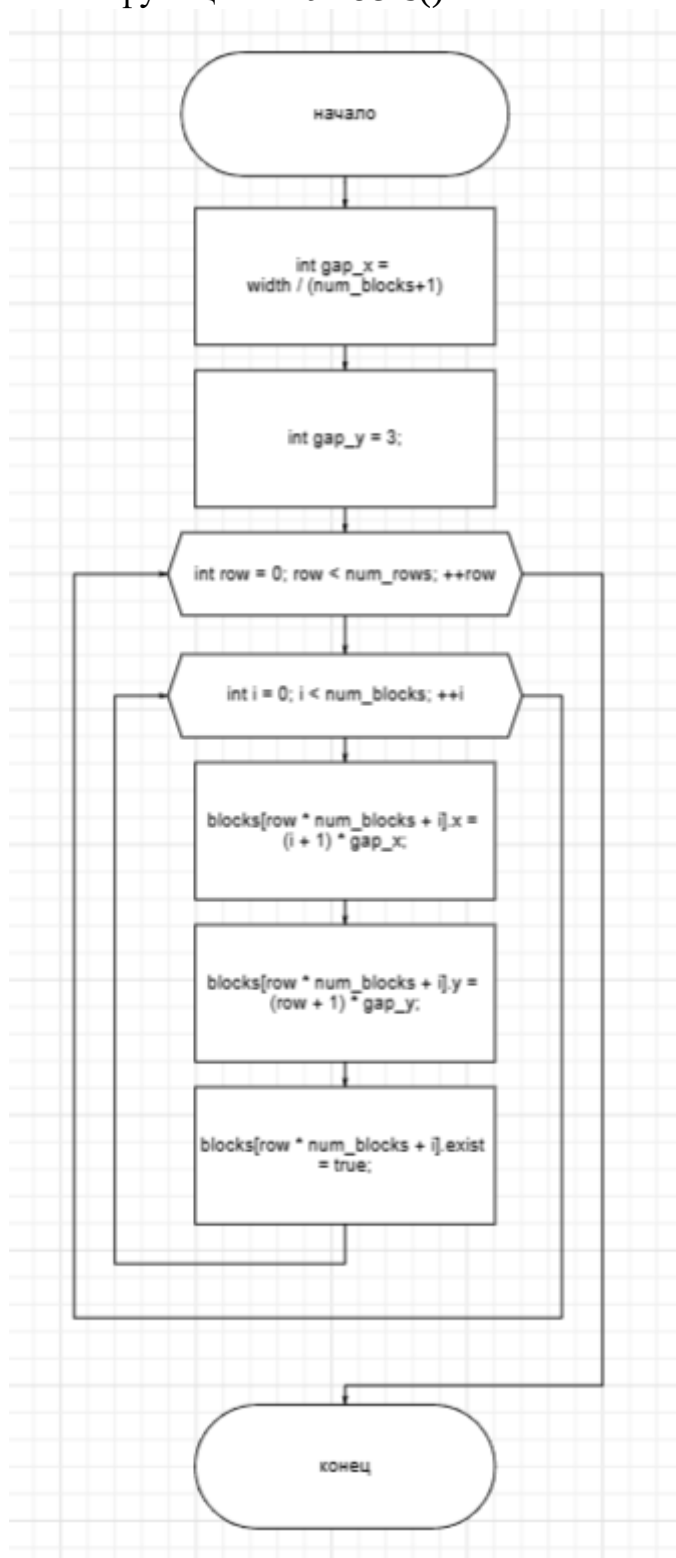


Схема функции moveBall(double x, double y)

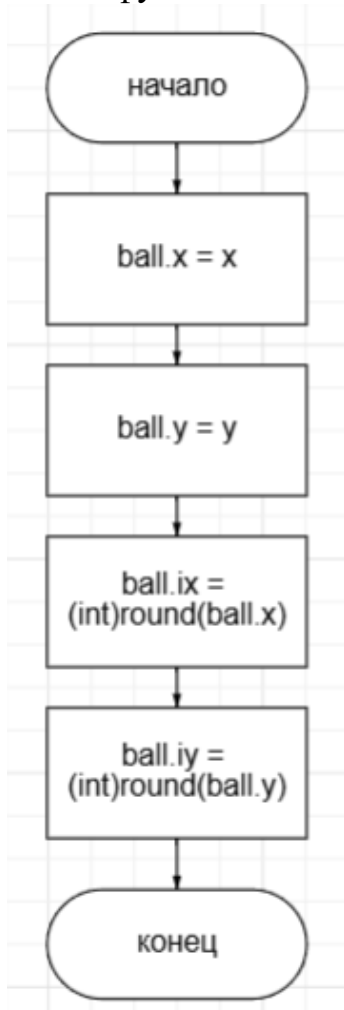


Схема функции init()

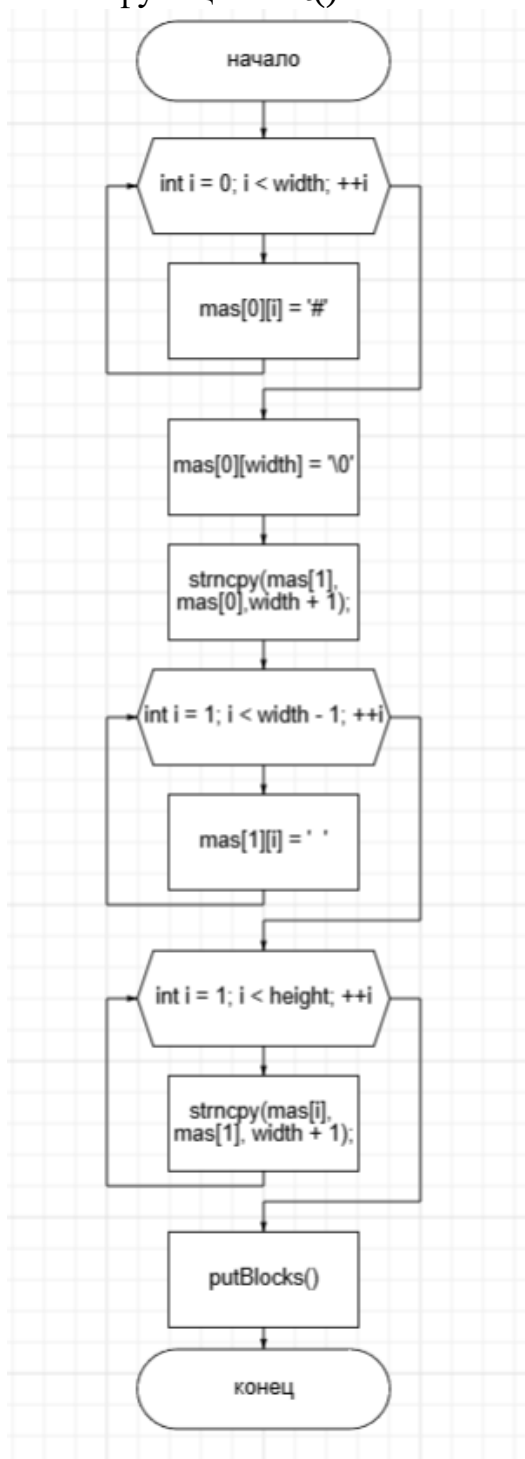


Схема функции putBall()

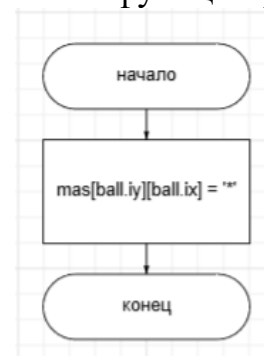


Схема функции show()

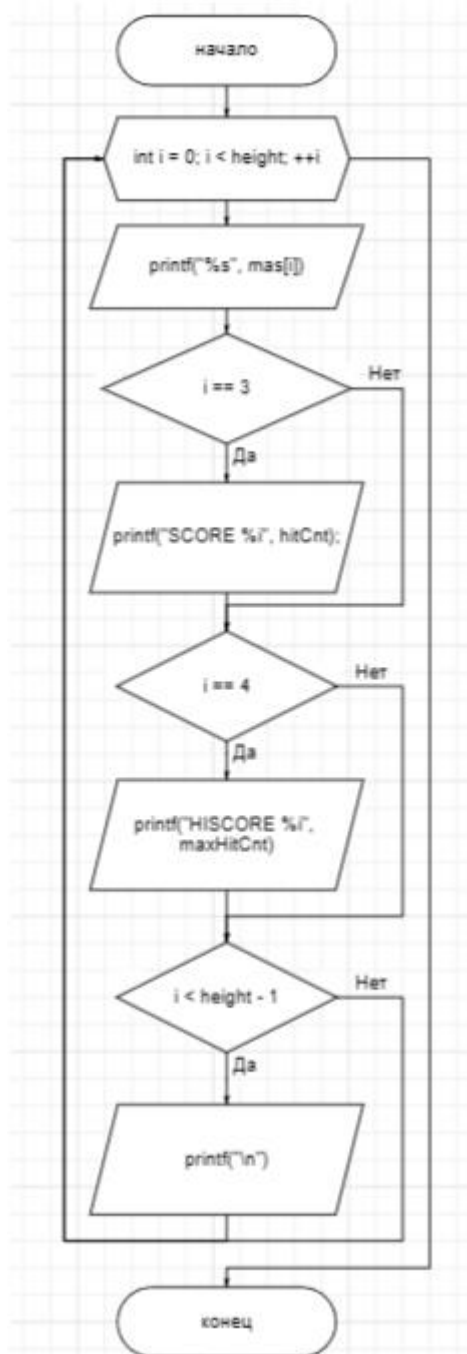


Схема функции putBlocks()

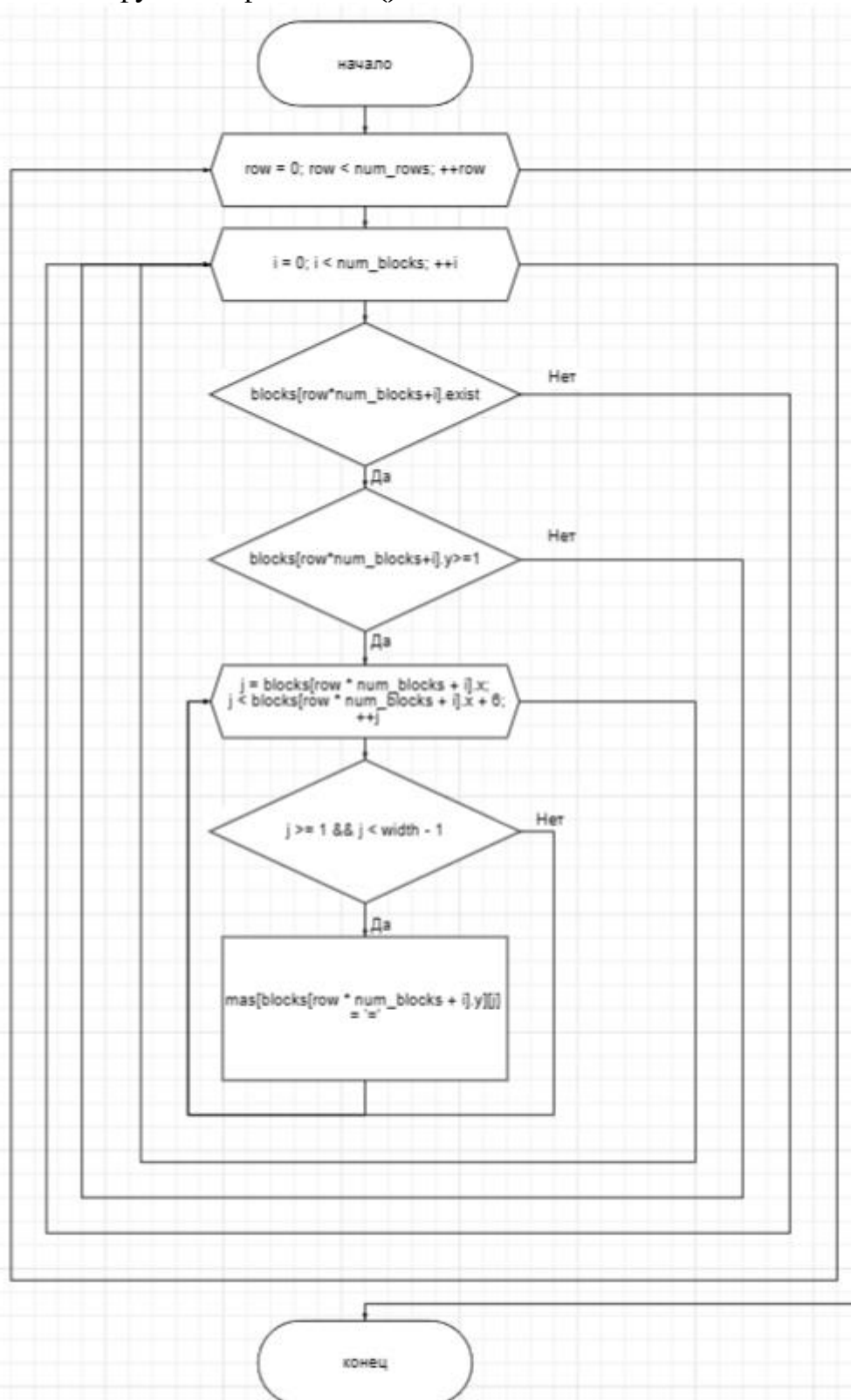


Схема функции putRacket()

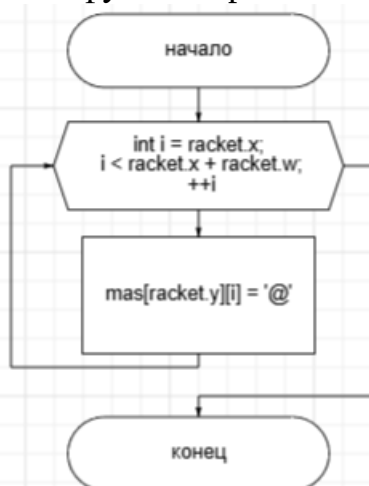


Схема функции moveRacket(int x)

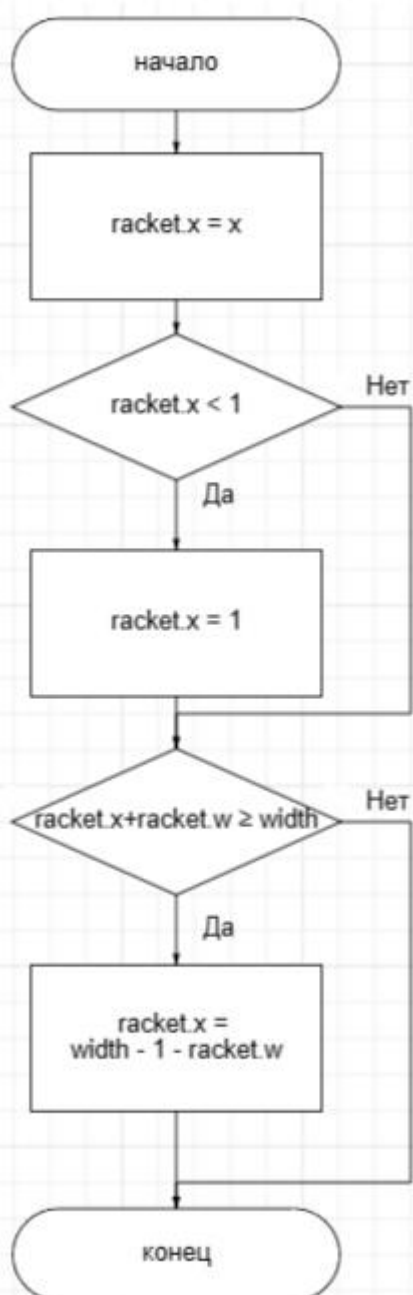


Схема функции setCur(int x, int y)

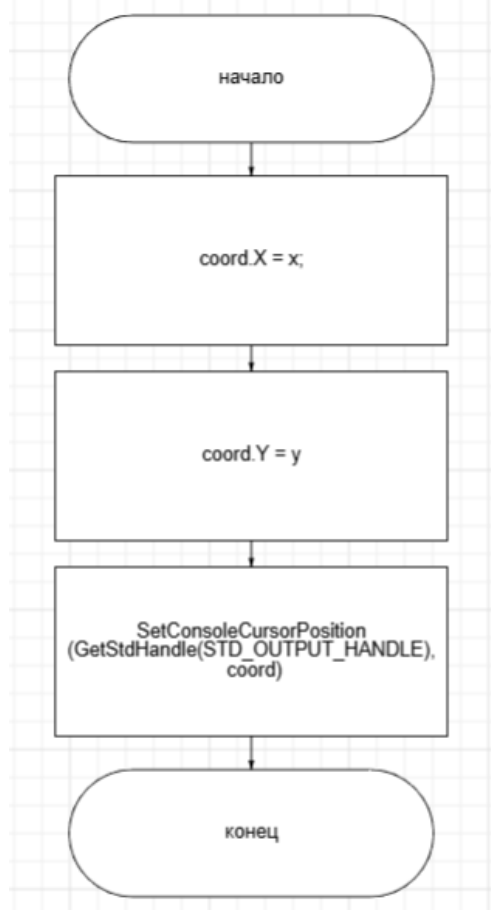


Схема функции compareScores(const void* a, const void* b)

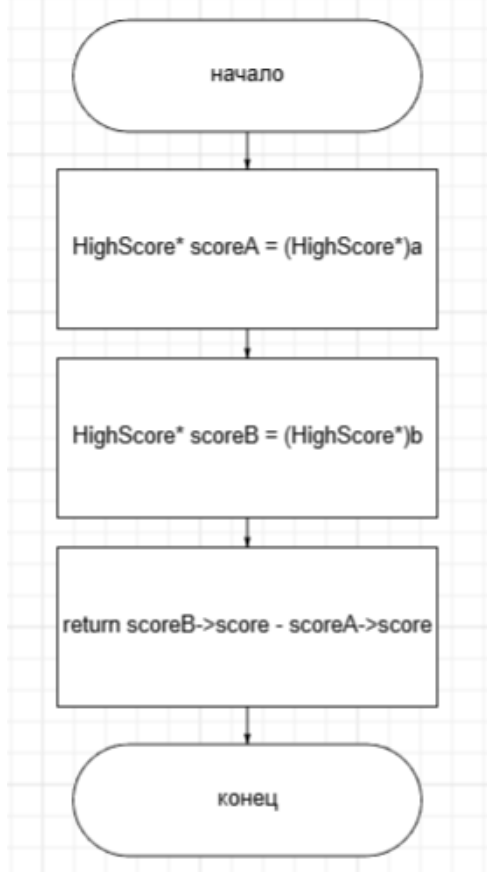


Схема функции displayHighScores()

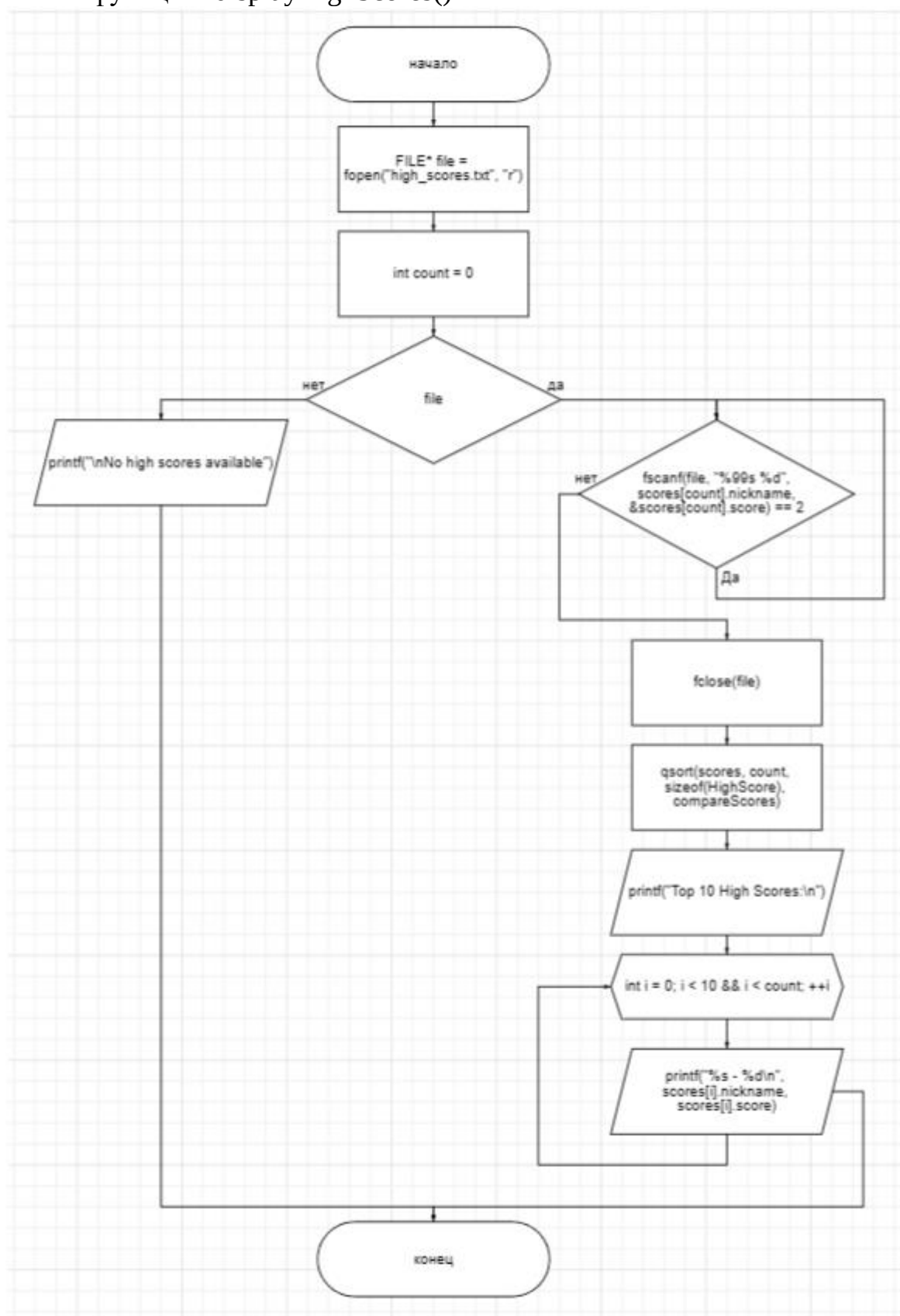


Схема функции `updateHighScores(const char* nickname, int score)`

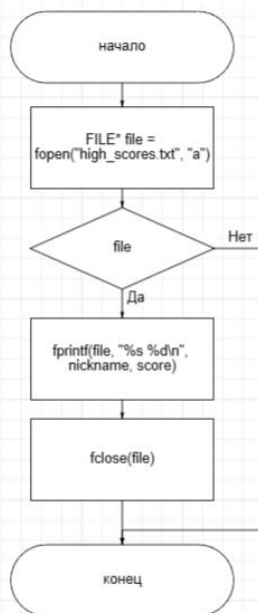
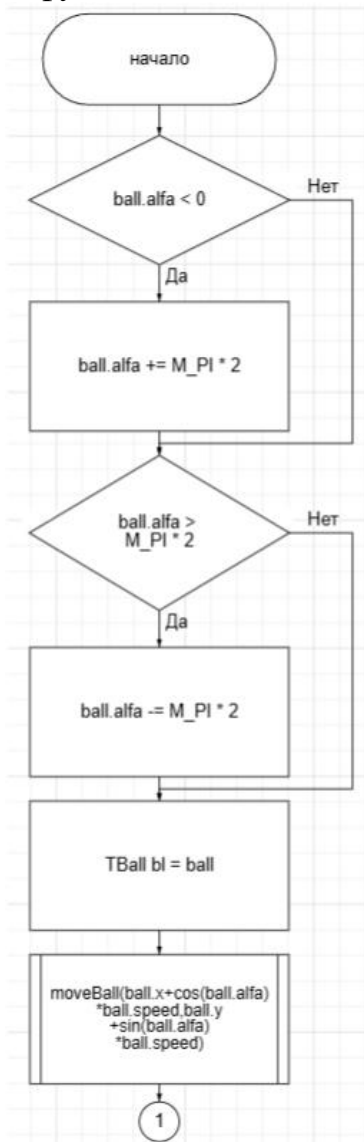
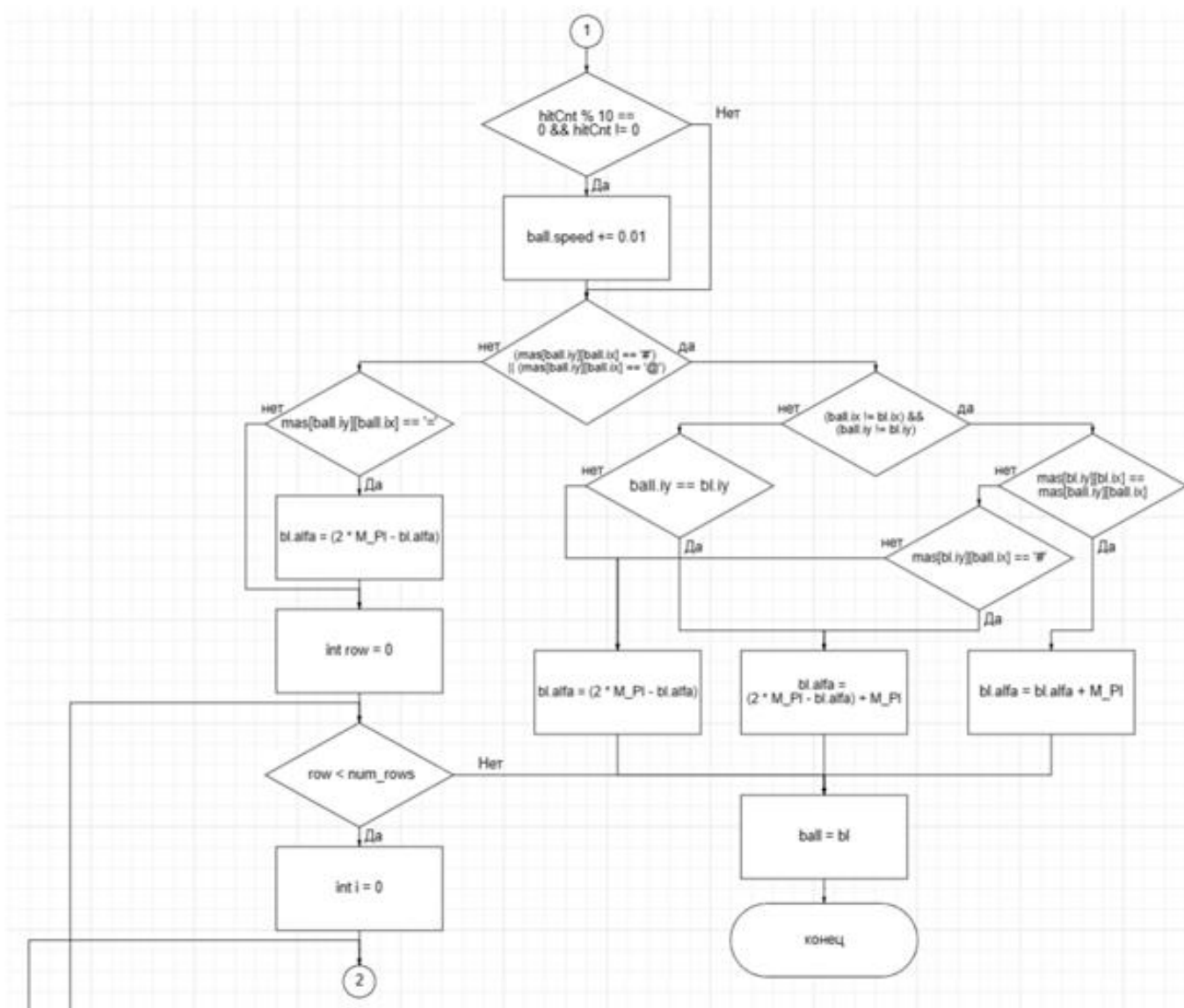
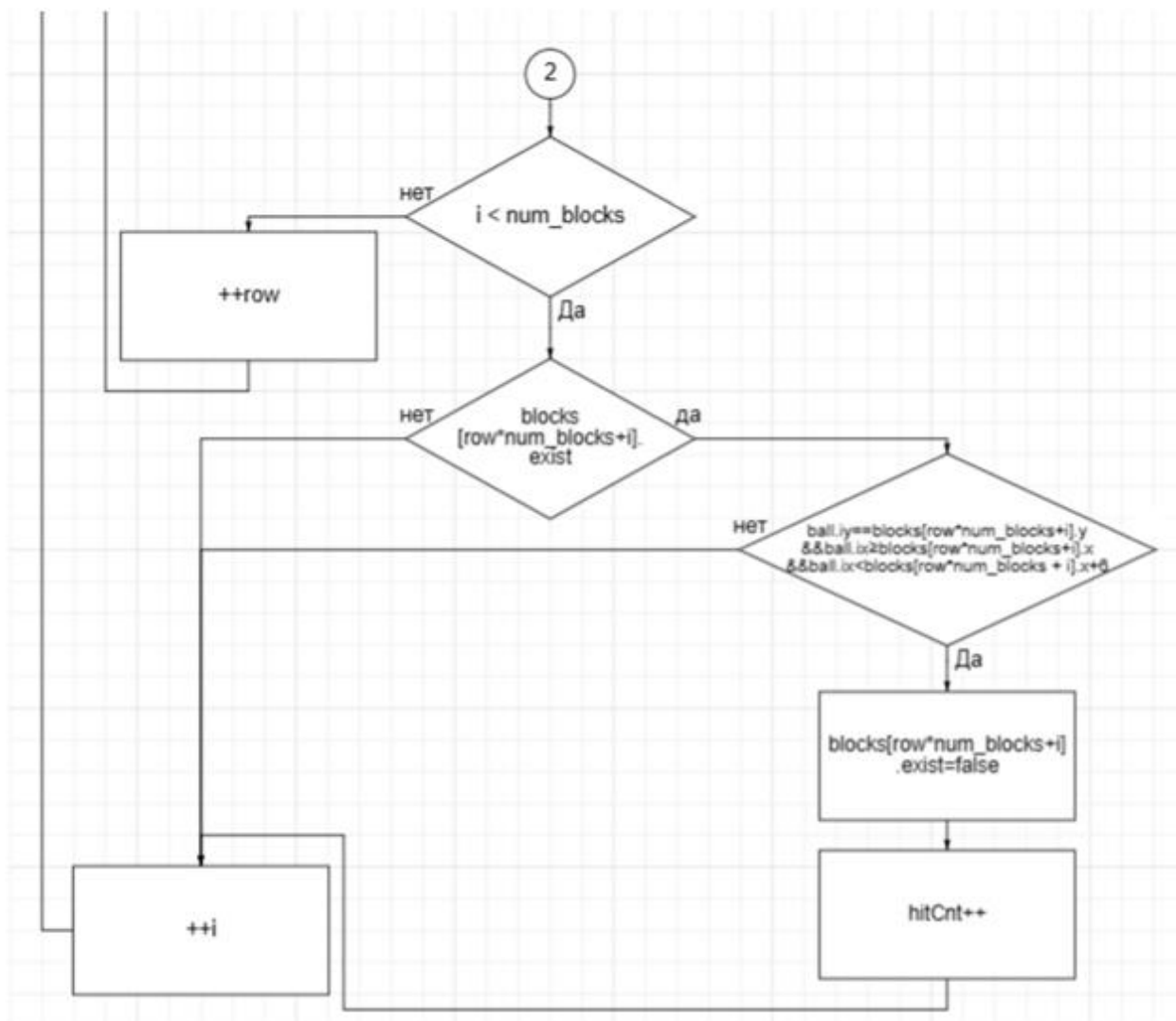


Схема функции `autoMoveBall()`







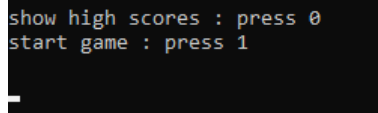
5. Отладка и тестирование

В качестве среды разработки была выбрана программа Microsoft Visual Studio 2019. Программа обладает всеми средствами необходимыми при разработке и отладке программы. Для отладки использовались несколько возможностей Visual Studio: точка останова, трассировка, анализ содержимого переменных. Тестирование проводилось во время разработки и также после завершения разработки. В ходе нее было выявлено огромное количество проблем, связанных с работой с файлами, работой с памятью, работой с полями структуры.

6. Руководство пользователя

Начало работы

1. На экране приветствия выберите "Начать игру" или "Просмотреть рекорды".



```
show high scores : press 0
start game : press 1
```

рисунок №1 – экран приветствия

Управление

- **A**: Переместить ракетку влево.
- **D**: Переместить ракетку вправо.
- **W**: Запустить мяч в начале игры или после потери жизни.
- **Escape**: Выход из игры.

Начало игры

- Игра начинается с ввода имени пользователя
- На всю игру дается три жизни.

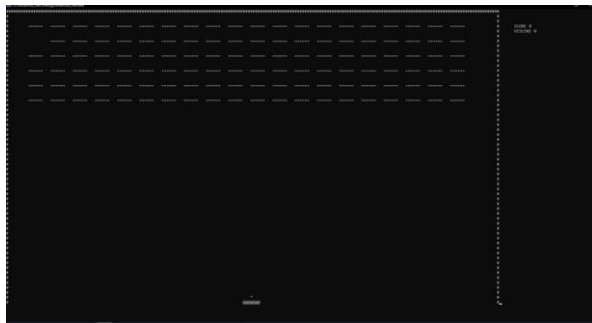
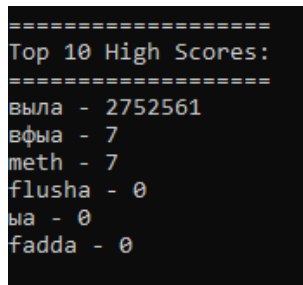


рисунок №2 – процесс игры

- Игра заканчивается, когда вы потеряете все жизни или нажмёте Escape.
- После завершения игры введенное вами имя пользователя автоматически сохранится в файл с рекордами

Просмотр рекордов

- в консоль выводится таблица с именами пользователей, набравших максимальное количество очков за игру



```
=====
Top 10 High Scores:
=====
выла - 2752561
вфыа - 7
meth - 7
flusha - 0
ыа - 0
fadda - 0
```

рисунок №3 – таблица рекордов

7. Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ. Была освоена работа с файлами, изучены функции работы с консолью и оперативной памятью. Также были получены основные навыки отладки и тестирования программ и программирования в среде Visual Studio 2019 на языках Си/C++ и Assembler. В рамках курсовой работы была написана игра Arkanoid. Она имеет все необходимые функции для комфортной игры.

В дальнейшем программу можно улучшить, путем добавления графического интерфейса, что позволит улучшить навигацию по меню программы, а так же улучшить визуальную часть игры

8. Список используемых источников

1. К.Н. Кинг “С программирование Современный подход” второе издание
2. Брайант Рэндал Э., О'Халларон Дэвид Р. “Компьютерные системы. Архитектура и программирование”
3. Р. Марек “Assembler на примерах. Базовый курс”
4. А. В. Михеенко “Геометрическая оптика”

Приложение А

Листинги программы

Файл main.cpp

```
#define _CRT_SECURE_NO_WARNINGS
#define _USE_MATH_DEFINES
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>
#include <windows.h>
#include <math.h>
#include <stdbool.h>
#include <time.h>
#include "ball.h"
#define width 200
#define height 60
#define num_blocks 20
#define num_rows 6
typedef struct {
    int x, y;
    bool exist;
} Tblock;
typedef struct {
    int x, y;
    int w;
} Tracket;
typedef struct {
    char nickname[100];
    int score;
} HighScore;
char mas[height][width + 1];
Tracket racket;
TBall ball;
Tblock blocks[num_blocks];
int hitCnt;
int maxHitCnt;
void autoMoveBall() {
    if (ball.alfa < 0) ball.alfa += M_PI * 2;
    if (ball.alfa > M_PI * 2) ball.alfa -= M_PI * 2;

    TBall b1 = ball;
    float deltaX, deltaY;
    __asm {
        fld ball.alfa
        fsincos
        fmul ball.speed
        fstp deltaX
        fmul ball.speed
```

```

        fstp deltaY
    }
    moveBall(ball.x + deltaX, ball.y - deltaY);
    if (hitCnt % 10 == 0 && hitCnt != 0) {
        ball.speed += 0.01;
    }

    if ((mas[ball.iy][ball.ix] == '#') || (mas[ball.iy][ball.ix] == '@')) {
        if ((ball.ix != bl.ix) && (ball.iy != bl.iy)) {
            if (mas[bl.iy][bl.ix] == mas[ball.iy][ball.ix]) {
                bl.alfa = bl.alfa + M_PI;
            }
            else {
                if (mas[bl.iy][ball.ix] == '#') {
                    bl.alfa = (2 * M_PI - bl.alfa) + M_PI;
                }
                else {
                    bl.alfa = (2 * M_PI - bl.alfa);
                }
            }
        }
        else if (ball.iy == bl.iy) {
            bl.alfa = (2 * M_PI - bl.alfa) + M_PI;
        }
        else {
            bl.alfa = (2 * M_PI - bl.alfa);
        }
        ball = bl;
    }
    else if (mas[ball.iy][ball.ix] == '=') {
        bl.alfa = (2 * M_PI - bl.alfa);
        for (int row = 0; row < num_rows; ++row) {
            for (int i = 0; i < num_blocks; ++i) {
                if (blocks[row * num_blocks + i].exist) {
                    if (ball.iy == blocks[row * num_blocks + i].y &&
ball.ix >= blocks[row * num_blocks + i].x && ball.ix < blocks[row *
num_blocks + i].x + 6) {
                        blocks[row * num_blocks + i].exist =
false;
                        hitCnt++;
                    }
                }
            }
        }
        ball = bl;
    }
}

void initRacket() {

```

```

    racket.w = 7;
    racket.x = (width - racket.w) / 2;
    racket.y = height - 1;
}
void putRacket() {
    for (int i = racket.x; i < racket.x + racket.w; ++i) {
        mas[racket.y][i] = '@';
    }
}
void initBlocks() {
    int gap_x = width / (num_blocks + 1);
    int gap_y = 3;
    for (int row = 0; row < num_rows; ++row) {
        for (int i = 0; i < num_blocks; ++i) {
            blocks[row * num_blocks + i].x = (i + 1) * gap_x;
            blocks[row * num_blocks + i].y = (row + 1) * gap_y;
            blocks[row * num_blocks + i].exist = true;
        }
    }
}

void putBlocks() {
    for (int row = 0; row < num_rows; ++row) {
        for (int i = 0; i < num_blocks; ++i) {
            if (blocks[row * num_blocks + i].exist) {
                if (blocks[row * num_blocks + i].y >= 1) {
                    for (int j = blocks[row * num_blocks + i].x; j <
blocks[row * num_blocks + i].x + 6; ++j) {
                        if (j >= 1 && j < width - 1) {
                            mas[blocks[row * num_blocks +
i].y][j] = '=';
                        }
                    }
                }
            }
        }
    }
}

void init() {
    for (int i = 0; i < width; ++i)
    {
        mas[0][i] = '#';
    }
    mas[0][width] = '\\0';
    strncpy(mas[1], mas[0], width + 1);
    for (int i = 1; i < width - 1; ++i)
    {

```

```

        mas[1][i] = ' ';
    }
    for (int i = 1; i < height; ++i)
    {
        strncpy(mas[i], mas[1], width + 1);
    }
    putBlocks();
}
void show() {
    for (int i = 0; i < height; ++i)
    {
        printf("%s", mas[i]);

        if (i == 3)
        {
            printf("        SCORE %i        ", hitCnt);
        }
        if (i == 4)
        {
            printf("        HISCORE %i        ", maxHitCnt);
        }
        if (i < height - 1)
        {
            printf("\n");
        }
    }
}
void moveRacket(int x)
{
    racket.x = x;
    if (racket.x < 1)
    {
        racket.x = 1;
    }
    if (racket.x + racket.w >= width)
    {
        racket.x = width - 1 - racket.w;
    }
}
void setCur(int x, int y)
{
    COORD coord;
    coord.X = x;
    coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);
}
void greeting()
{

```



```

int main() {
    int lives = 3;
    greeting();
    Sleep(2500);
    system("cls");
    printf("\nshow high scores : press 0");
    printf("\nstart game : press 1\n\n");
    char temp = _getch();
    if (temp == '1') {
        system("cls");
        char nickname[50];
        printf("\n\n\n\n\n\n\n\nWelcome to the ARKANOID game!\n");
        printf("Enter your nickname: ");
        scanf("%s", nickname);
        bool run = false;
        initRacket();
        initBall();
        initBlocks();
        hitCnt = 0;
        maxHitCnt = 0;
        do {
            setCur(0, 0);
            if (run)
            {
                autoMoveBall();
            }
            if (ball.iy > height)
            {
                lives--;
                if (lives > 0) {
                    run = false;
                    moveBall(racket.x + racket.w / 2, racket.y - 1);
                }
                else {
                    printf("Game Over!\n");
                    break;
                }
                hitCnt = 0;
            }
            else if (hitCnt > maxHitCnt)
            {
                maxHitCnt = hitCnt;
            }
            init();
            putRacket();
            putBall();
            show();
            if (GetKeyState('A') < 0) moveRacket(racket.x - 1);

```

```

        if (GetKeyState('D') < 0) moveRacket(racket.x + 1);
        if (GetKeyState('W') < 0) run = true;
        if (!run)
        {
            moveBall(racket.x + racket.w / 2, racket.y - 1);
        }
        Sleep(5);
    } while (GetAsyncKeyState(VK_ESCAPE) >= 0);
    updateHighScores(nickname, maxHitCnt);
}
if (temp == '0') {
    displayHighScores();
}
return 0;
}

```

Файл ball.cpp

```

#include "ball.h"
extern char mas[][200 + 1];
extern TBall ball;
void putBall() {
    mas[ball.iy][ball.ix] = '*';
}
void moveBall(double x, double y) {
    ball.x = x;
    ball.y = y;
    ball.ix = (int)round(ball.x);
    ball.iy = (int)round(ball.y);
}
void initBall() {
    moveBall(2, 2);
    ball.alfa = -1;
    ball.speed = 0.5;
}

```

Файл ball.h

```

#ifndef BALL_H
#define BALL_H
#include <math.h>
typedef struct {
    double x, y;
    int ix, iy;
    double alfa;
    double speed;
} TBall;
void putBall();
void moveBall(double x, double y);
void initBall();
#endif // BALL_H

```

Приложение В

Результаты работы программы



рисунок №4 – приветствие



рисунок №5 – процесс игры

```
=====
Top 10 High Scores:
=====
выла - 2752561
meth - 7
вфыа - 7
gamer1 - 1
flusha - 0
ыа - 0
fadda - 0
```

рисунок №6 – вывод таблицы рекордов

Приложение С

Результаты работы программы

Файл high_scores.txt

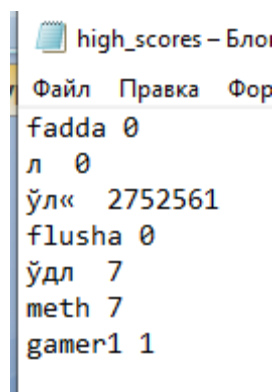


рисунок №7 –файл, где сохраняются имена пользователей и их рекорды