

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение  
высшего образования

«КРЫМСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ им. В. И. ВЕРНАДСКОГО»  
ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ

Кафедра компьютерной инженерии и моделирования

**Разработка многопользовательской игры "Тетрис"**

Курсовая работа

по дисциплине «Программирование»

студента 1 курса группы ПИ-б-о-201

Юращика Николая Александровича

направления подготовки 09.03.04 «Программная инженерия»

Научный руководитель

старший преподаватель кафедры

компьютерной инженерии и моделирования

\_\_\_\_\_  
(оценка)

\_\_\_\_\_  
Чабанов В.В.

(подпись, дата)

Симферополь, 2021

## РЕФЕРАТ

Разработка многопользовательской игры "Тетрис". – Симферополь: ФТИ КФУ им. В. И. Вернадского, 2021. – 41 с., 17 ил., 7 ист.

Объектом исследования данной работы является игра «Тетрис».

Цель работы – создание рабочего игрового проекта с реализацией клиент-серверной системы. Сервер реализуется с использованием языка Python, а клиент с использованием языка C++.

Актуальность работы обусловлена полным отсутствием аналогов на рынке, так как под «многопользовательским тетрисом» обычно понимают просто игру на выбывание, где у каждого игрока своё игровое поле и игра продолжается до тех пор, пока один из игроков не достигнет верхней границы – проиграет. Данный же проект реализует игру двух игроков на одном поле с последовательным управлением падающих фигур.

Было рассмотрено многочисленное количество вариантов реализации клиент-серверных систем, в ходе которых было принято решение использовать C++ для написания клиента и Python – для сервера, а для их взаимодействия использовать сокеты, что позволило обеспечить обмен данными в режиме достаточно приближенному к реальному времени.

ПРОГРАММИРОВАНИЕ, C++, SFML, PYTHON, СОКЕТЫ, КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ

## ОГЛАВЛЕНИЕ

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ .....	4
ВВЕДЕНИЕ.....	5
ГЛАВА 1 ПОСТАНОВКА ЗАДАЧИ.....	7
1.1 Цель проекта .....	7
1.2 Существующие аналоги .....	7
1.3 Основные отличия от аналогов .....	7
1.4 Техническое задание.....	8
ГЛАВА 2 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ.....	11
2.1 Анализ инструментальных средств.....	11
2.2 Описание алгоритмов .....	12
2.2.1 Алгоритм работы окон .....	12
2.2.2 Алгоритм проверки логики игры .....	15
2.3 Описание структур данных.....	16
2.3.1 Клиент .....	16
2.3.2 Сервер.....	17
2.4 Описание основных модулей.....	18
ГЛАВА 3 ТЕСТИРОВАНИЕ ПРОГРАММЫ.....	23
3.1 Тестирование исходного кода.....	23
3.2 Тестирование интерфейса пользователя и юзабилити.....	23
ГЛАВА 4 ПЕРСПЕКТИВЫ ДАЛЬНЕЙШЕГО РАЗВИТИЯ ПРОЕКТА .....	25
4.1 Перспективы технического развития.....	25
4.2 Перспективы монетизации.....	25
ЗАКЛЮЧЕНИЕ .....	26
ЛИТЕРАТУРА.....	27

**СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ**

ПО	Программное обеспечение
IDE	Интегрированная среда разработки
TCP	Transmission Control Protocol
IP	Internet Protocol
SFML	Simple and Fast Multimedia Library

## ВВЕДЕНИЕ

Игра «Тетрис» ориентирована на широкий возрастной диапазон пользователей. Согласно многим исследованиям, «Тетрис» является комплексной задачей для мозга. Он затрагивает множество когнитивных процессов: внимание, координацию между руками и зрением, память и пространственное восприятие.

Данный проект выводит старую игру на новый уровень, ведь теперь, после каждого Вашего хода, следующей фигурой – так называемой тетрамино – будет управлять другой человек. При таком подходе старые тактики уже не принесут должного эффекта и придётся искать новые подходы. Теперь продолжительность игры зависит не только от Ваших решений, но и от решений соперника, хотя в данном случае правильнее сказать напарника. Ведь в конце нет ни победителя, ни проигравшего, каждый игрок получает столько баллов, сколько линий он смог собрать.

Для реализации данного проекта были выбраны такие языки программирования как C++ и Python.

При планировании данного проекта была поставлена следующая цель – реализовать клиент-серверный вариант игры «Тетрис».

Для достижения цели в данной работе были поставлены следующие задачи:

1. изучить «старые» правила игры и обновить их для использования в многопользовательском варианте;
2. изучить библиотеку SFML для языка программирования C++;
3. изучить возможности языка SQL, установку СУБД MySQL, а также библиотеку для работы с MySQL на Python - pymysql;
4. изучить работу с сокетами;
5. реализация программного кода клиента и сервера;
6. отладка и тестирование.

Объектом исследования является игра «Тетрис».

Предметом исследования является реализация клиент-серверного многопользовательского варианта игры «Тетрис».

Методы исследования:

- моделирование – построение и изучение моделей с целью приобретения новых знаний, улучшение характеристик объектов исследований или управление ими;
- сравнение – сравнение предметов между собой на выявление сходства и различия, преимуществ и недостатков.

## ГЛАВА 1

### ПОСТАНОВКА ЗАДАЧИ

#### 1.1 Цель проекта

По итогу разработки планируется создать многопользовательский вариант игры «Тетрис» для двух игроков. Игра будет отличаться правилами – теперь на одном поле будут поочерёдно ходить оба игрока, что сделает бессмысленным использование старых тактик и заставит пользователей адаптироваться и находить новые стратегии для успешной игры.

#### 1.2 Существующие аналоги

Прямых аналогов данного проекта не существует, есть лишь варианты, где у каждого игрока своё поле, а игра продолжается пока один из них не достигнет верхнего предела, то есть проиграет. В пример можно привести такие проекты, как:

- min2win Тетрис – браузерная онлайн версия классического тетриса с возможностью игры вдвоём на разных полях. Нет ведения статистики после покидания страницы с игрой;
- Playmap Тетрис на двоих – похожая на предыдущий проект реализация, однако правила игры изменены – при собирании линии на одном поле, на втором образуются лишние фигуры снизу, которые поднимают основные наверх, что значительно усложняет игру. Также нет ведения статистики игроков после покидания страницы и неудобное управление, которое при входе необходимо изменять, что создаёт неудобства;
- NoorGame Tetris – классический тетрис без особых нововведений, браузерный проект.

#### 1.3 Основные отличия от аналогов

- Наличие сервера для ведения статистики игроков;

- Возможность получения монет за игру для осуществления покупок во встроенном магазине (например, текстур);
- Многопользовательский вариант исполнения для игры на одном поле.

#### 1.4 Техническое задание

Программа должна соответствовать следующему техническому заданию:

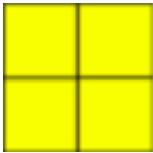
1. Главное меню программы должно содержать несколько кнопок для:
  - 1.1. Перехода в подменю с игровыми комнатами, с возможностью подключения к уже существующей, а также создания собственной комнаты;
  - 1.2. Перехода в подменю «Магазин», с возможностью покупки новых текстур и их последующей установки;
  - 1.3. Перехода в подменю «Статистика» для просмотра топ 10 ведущих по очкам игроков;
  - 1.4. Перехода в подменю «Помощь».
2. Программа должна корректно интерпретировать и соблюдать следующие основные правила игры.

Для начала игры необходимо два игрока: один создаёт комнату, а второй подключается к ней. Право первого хода определяется случайным образом и не зависит от игроков и их действий.


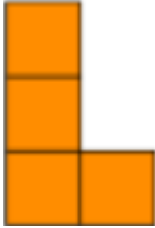
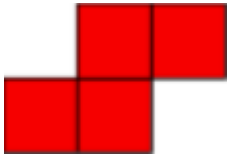

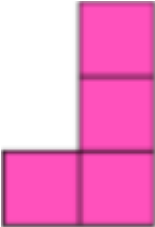
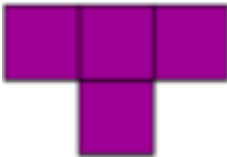
После начала игры появляется игровое поле с размерами 20 на 10 клеток, а также первая фигура – тетрамино.

Тетрамино может быть 7 видов, описанных в программе (таблица 1.1).

Таблица 1.1. Возможные виды тетрамино

O	
---	---



I	
L	
S	
Z	
J	
T	

Тетрамино опускается вниз на одну клетку каждые 0.5 секунды. Управление осуществляется с помощью стрелок вверх, вправо и влево тем игроком, который в данный момент обладает правом хода. Клавиши «влево» и «вправо» соответственно меняют положение тетрамино по горизонтали. Клавиша «вверх»

осуществляет вращение фигуры относительно ее центра. При этом тетрамино не может выходить за пределы игрового поля. Когда оно касается боковых граней последующее смещение в сторону края не происходит. Если фигура касается нижней границы игрового поля или любой другой фигуры, которая уже упала, то она считается упавшей и управление ей более невозможно. Право хода передаётся другому игроку, а также создаётся новая фигура сверху, которая аналогично начинает сдвигаться вниз на клетку по таймеру.

Целью каждого игрока является собирание как можно большего количества линий – когда тетрамино полностью заполняют горизонтальную линию, то есть все 10 клеток. При этом сама линия удаляется, а все фигуры, которые были выше нее, опускаются вниз на одну клетку. Игрок, у которого на данный момент было право хода, получает 1 балл за каждую собранную линию.

Игра заканчивается на моменте, когда какая-либо тетрамино после падения пересекает верхнюю границу поля. При этом каждому игроку начисляется то количество очков и монет, сколько баллов они накопили за данную игру. После окончания игры игроков переносит в главное меню.

**ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ****2.1 Анализ инструментальных средств**

Одной из самых масштабных задач в подобных проектах является реализация графической составляющей. В данном решении необходимо описать: главное меню, меню комнат, меню магазина, меню статистики, меню помощи, игровое поле и непосредственно тетрамино. Для этого было решено использовать библиотеку SFML – Simple and Fast Multimedia Library – это библиотека для представления мультимедийных данных. SFML обеспечивает простой интерфейс для разработки игр и прочих мультимедийных приложений. Состоит она из пяти модулей:

- system – управление временем и потоками, он является обязательным, так как все модули зависят от него;
- window – управление окнами и взаимодействием с пользователем;
- graphics – делает простым отображение графических примитивов и изображений;
- audio – предоставляет интерфейс для управления звуком;
- network – для сетевых приложений.

Из аналогов были также рассмотрены такие библиотеки как SDL и Allegro. Allegro – это кроссплатформенная библиотека, в основном предназначенная для видеоигр и мультимедийных программ. Она выполняет общие, низкоуровневые задачи, такие как создание окон, прием пользовательского ввода, загрузка данных, рисование изображений, воспроизведение звуков. SDL – это свободная кроссплатформенная мультимедийная библиотека, реализующая единый программный интерфейс к графической подсистеме, звуковым устройствам и средствам ввода для широкого спектра платформ.

Несмотря на то, что выбор всё равно остался за SFML, у данного решения есть некоторые недостатки, например, невозможность создания поля для ввода, которое придётся создавать другими средствами.

Также был выбран ряд других инструментов, таких как:

- Среда Microsoft Visual Studio 2019 – была выбрана в пользу знакомого интерфейса, возможности создания проекта с использованием языка C++ и подключения библиотеки SFML;
- Текстовый редактор Sublime Text 3 для редактирования кода на языке Python;
- Среда IDLE для отладки кода на языке Python – была выбрана так как идет «в комплекте» со стандартной установкой Python в ОС Windows;
- C++ и Python – как изучаемые языки программирования;
- Модуль WinSock2 – сетевое взаимодействие со стороны клиента;
- Библиотека socket – сетевое взаимодействие со стороны сервера;
- Библиотека PyMySQL для работы с СУБД MySQL в Python.

## **2.2 Описание алгоритмов**

### **2.2.1 Алгоритм работы окон**

В данном проекте присутствуют понятия окон и перехода между ними. В оконной логике приложения есть 4 основных блока:

1. Блок управления – основной цикл, в котором последовательно строятся основные окна приложения в зависимости от готовности перейти на следующий блок;
2. Блок авторизации/регистрации – первый оконный блок приложения, в котором можно зарегистрироваться или войти в существующий аккаунт. Если вход возможен, то осуществляется переход на следующий блок;
3. Блок главное меню – второй блок, содержащий в себе кнопки:
  - 3.1.«Играть» – отправляет запрос на сервер для формирования открытых комнат и осуществляет переход на следующий блок – подменю комнат;
  - 3.2.«Магазин» – отправляет запрос на сервер для формирования доступных текстур и осуществляет переход на блок – магазин;
  - 3.3.«Статистика» – отправляет запрос на сервер для формирования десяти лучших игроков и отображает их;

- 3.4. «Помощь» – отображает краткую информацию о проекте.
4. Блок меню комнат – подменю для создания собственной комнаты и подключения к уже существующим, содержит в себе кнопки:
- 4.1. «Создать игру» - отправляет запрос о создании новой комнаты и циклично проверяет наличие второго игрока, после чего осуществляет переход к блоку игрового поля;
- 4.2. Таблица комнат – таблица, содержащая в себе номер комнаты и имя игрока, который ее создал, при нажатии на любую запоминается ее номер;
- 4.3. «Подключиться» - отправляет запрос о подключении к выбранной комнате на сервер, после чего осуществляет переход к блоку игрового поля.
5. Блок магазина – подменю для покупки и установки текстур для тетрамино, содержит в себе таблицу текстур, а также кнопки:
- 5.1. «Купить» – отправляет запрос на сервер, где определяется возможность покупки выбранной текстуры, после получения положительного ответа делает эту текстуру возможной для установки;
- 5.2. «Установить» – проверяет куплена ли текстура, в зависимости от проверки, если положительная – делает данную текстуру активной, если отрицательная – ничего не изменяет.
6. Блок игрового поля – отображение игрового поля, падающих фигурок тетрамино. Право хода определяется сервером в случайном порядке. Игрок, который ходит, осуществляет управление фигурой и после каждого ее падения на одну клетку отправляет данные на сервер для синхронизации второго клиента. После окончания хода отправляет информацию об этом на сервер, чтобы ожидающий игрок синхронизировался и начал свой ход.

При запуске программы появляется окно входа и регистрации пользователя. Пользователь вводит данные логин и пароль, клиент проверяет все ли поля заполнены корректно, после этого отправляет запрос на сервер для проверки существования учетной записи данного пользователя (рисунок 2.1).

```

bool logining(string login) {
    int login_success = 0;
    send(Connection, login.c_str(), login.length(), NULL);
    char msg[64];
    recv(Connection, msg, sizeof(msg), NULL);
    try {
        std::string msg_str = decoding(msg);
        std::string sep = ":";
        int i = 0;
        size_t pos = 0;
        std::string token;
        while ((pos = msg_str.find(sep)) != std::string::npos) {
            token = msg_str.substr(0, pos);
            if ((i == 0) && (login_success == 0)) {
                if (token == "success") {
                    login_success = 1;
                }
                else {
                    break;
                }
            }
            else {
                if (i == 1) { name = token; }
                if (i == 2) { money = token; }
                if (i == 3) { score = token; }
            }
            i++;
            msg_str.erase(0, pos + sep.length());
        }
        user_textures = msg_str;
        if (login_success == 1) {
            return 1;
        }
        else {
            return 0;
        }
    }
    catch (const std::exception& e) {
        return 0;
    }
}

```

Рисунок 2.1. Проверка авторизации или регистрация пользователя.

После этого сервер проверяет данные в базе данных и возвращает результат (рисунок 2.2). Если такая учетная запись есть в базе данных пользователь попадет в игровое меню, а если ее не существует, тогда пользователю придется зарегистрироваться.

```

def authorization(user, swap):
    this.swap = swap
    data = user.recv(2048)
    msg = data.decode('utf-8')
    msg = msg.split(":")
    if msg[0] == 'login':
        print(f'login: {msg}')
        login = msg[1]
        password = msg[2]
        user_param = check_user(login, password)
        if user_param:
            msg = 'success:' + str(user_param['login']) + ":" +
                str(user_param['money']) + ":" + str(user_param['score']) +
                ":" + str(user_param['textures'])
            msg = str(len(msg)) + msg
            print(f'Login success: {msg}')
            user.send(msg.encode('utf-8'))
        else:
            text = 'fail'
            msg = str(len(text)) + text;
            user.send(msg.encode('utf-8'))
            authorization(user, this.swap)
    elif msg[0] == 'reg':
        print(f'reg: {msg}')
        login = msg[1]
        password = msg[2]
        user_param = create_user(login, password)
        if user_param:
            msg = 'success:' + str(user_param['login']) + ":" +
                str(user_param['money']) + ":" + str(user_param['score']) +
                ":" + str(user_param['textures'])
            msg = str(len(msg)) + msg
            print(f'Login success: {msg}')
            user.send(msg.encode('utf-8'))

```

Рисунок 2.2. Получение данных об авторизации или регистрации на сервере и их обработка.

### 2.2.2 Алгоритм проверки логики игры

После создания комнаты происходит запись в базу данных в таблицу rooms об имени пользователя и номере комнаты. Далее, после подключения второго игрока к комнате, запись обновляется – дописывается имя второго игрока, а также создается запись во временном хранилище сервера. Запись в базе данных нужна для синхронизации временного хранилища на обоих потоках. Во временном хранилище находится информация о праве хода, а также текущее положение

тетрамино. После выполнения всех вышеперечисленных действий на обоих потоках запускается функция `game` (рисунок 2.3), отвечающая за приём и передачу информации о положении тетрамино, праве хода, а также синхронизацию между двумя потоками с помощью временного хранилища.

```
def game(user, user_param, swap):
    global id_s
    this.swap = swap
    k = this.swap[str(id_s)][0]
    # 0,1 - для решения чей ход
    # 2   - для смены ходящего
    # 3   - для окончания игры

    while k != 3:

        msg = str(this.swap[str(id_s)][0]) + '|' + str(this.swap[str(id_s)][1])
        msg = str(len(msg)) + msg
        user.send(msg.encode('utf-8'))
        data = user.recv(2048)
        msg = data.decode('utf-8')
        if msg[0] == this.swap[str(id_s)][0]:
            this.swap[str(id_s)] = [indicator, msg[2:]]
        elif msg[0] == "2":
            this.swap[str(id_s)] = [opponent, msg[2:]]
        elif msg[0] == "3":
            msg = msg.split(':')
            score = int(msg[1])
            this.swap[str(id_s)] = ["e", "0|0|0|0|0|0|0|0"]
            msg = end_of_game(user_param, score, id_s)
            user.send(msg.encode('utf-8'))
            k = 3
```

Рисунок 2.3. Функция `game`.

## 2.3 Описание структур данных

### 2.3.1 Клиент

Главными структурами данных выступают:

1. Двумерный массив `fields`, хранящий в себе игровое поле с фигурами в виде чисел.
2. Двумерный массив видов тетрамино (рисунок 2.4);
3. Структура `Point`, описывающая текущий вид и местоположение падающей фигуры на поле (рисунок 2.5).



```
int figures[7][4] =
{
    1,3,5,7, // I
    2,4,5,7, // Z
    3,5,4,6, // S
    3,5,4,7, // T
    2,3,5,7, // L
    3,5,7,6, // J
    2,3,4,5, // O
};
```

Рисунок 2.4. Двумерный массив видов тетрамино.

```
struct Point
{
    int x, y;
} a[4], b[4];
```

Рисунок 2.5. Структура Point.

К структуре Point обращается большинство функций логики игры. При каждом падении тетрамино на клетку вниз, а также изменении горизонтального положения, происходит изменение координат в структуре. Далее эти данные передаются на сервер для синхронизации второго клиента.

Запись координат в двумерный массив fields происходит только после падения фигуры. В этом случае выполняется проверка, касается ли фигура нижней границы поля или другой фигуры снизу, после чего происходит запись координат в массив поля, а также отправка данных на сервер, с указанием окончания хода.

Двумерный массив видов тетрамино используется для формирования новой фигуры в начале хода.

### 2.3.2 Сервер

Для хранения информации об аккаунтах пользователей используется база данных под управлением MySQL. В базе данных присутствуют таблицы «users», «rooms» и «shop». Таблица «users» хранит в себе имя пользователя, пароль, статус, рейтинг, количество монет, а также приобретенные текстуры (рисунок 2.6). Таблица «rooms» хранит информацию о созданных комнатах, их номер, логин

первого игрока, логин второго игрока и статус комнаты (рисунок 2.7). Таблица «shop» хранит информацию о доступных для приобретения текстур, а также их стоимости (рисунок 2.8).

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	login	text	utf8_general_ci		Нет	Нем		
2	password	text	utf8_general_ci		Нет	Нем		
3	find_game	text	utf8_general_ci		Нет	Нем		
4	score	int(11)			Нет	Нем		
5	money	int(11)			Нет	Нем		
6	queue	text	utf8_general_ci		Нет	Нем		
7	num 	int(5)			Нет	Нем		AUTO_INCREMENT
8	textures	text	utf8_general_ci		Нет	Нем		

Рисунок 2.6. Структура таблицы «users».

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id 	int(3)			Нет	Нем		AUTO_INCREMENT
2	user1	text	utf8_general_ci		Нет	Нем		
3	user2	text	utf8_general_ci		Да	NULL		
4	status	text	utf8_general_ci		Нет	Нем		

Рисунок 2.7. Структура таблицы «rooms».


#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	num 	int(11)			Нет	Нем		AUTO_INCREMENT
2	link	text	utf8_general_ci		Нет	Нем		
3	price	int(11)			Нет	Нем		

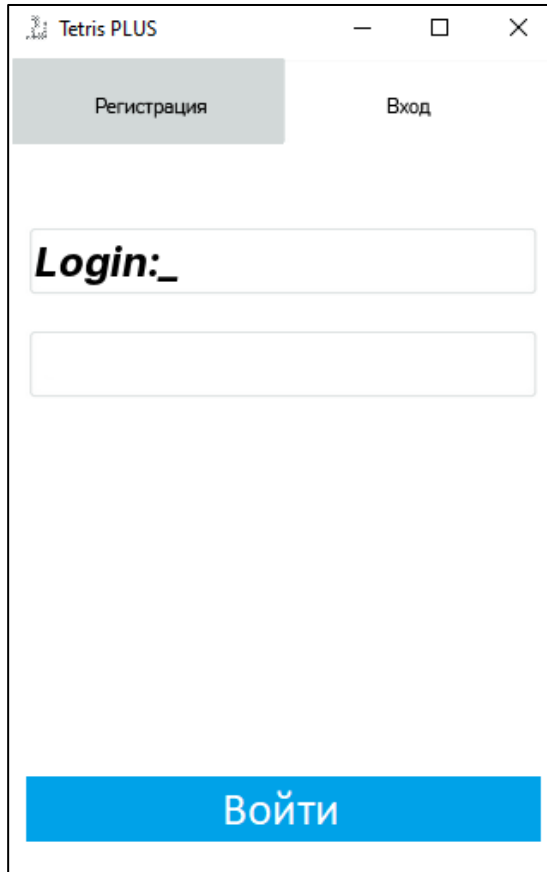
Рисунок 2.8. Структура таблицы «shop».

## 2.4 Описание основных модулей

После запуска игры пользователь попадает в окно авторизации (рисунок 2.9). После входа или регистрации нового аккаунта открывается главное меню (рисунок 2.10). Из главного меню пользователь может:

- перейти в подменю помощи нажав на кнопку «Помощь» (рисунок 2.11);

- перейти в подменю статистики топ 10 игроков нажав на кнопку «Статистика» (рисунок 2.12);
- перейти в подменю магазина нажав на кнопку «Магазин» (рисунок 2.13);
- перейти в подменю комнат нажав на кнопку «Играть» (рисунок 2.14);



The image shows a web application window titled "Tetris PLUS". At the top, there are two buttons: "Регистрация" (Registration) and "Вход" (Login). Below these, there is a text input field with the label "Login:\_" and a password input field. At the bottom of the window, there is a large blue button labeled "Войти" (Login).

Рисунок 2.9. Окно авторизации.

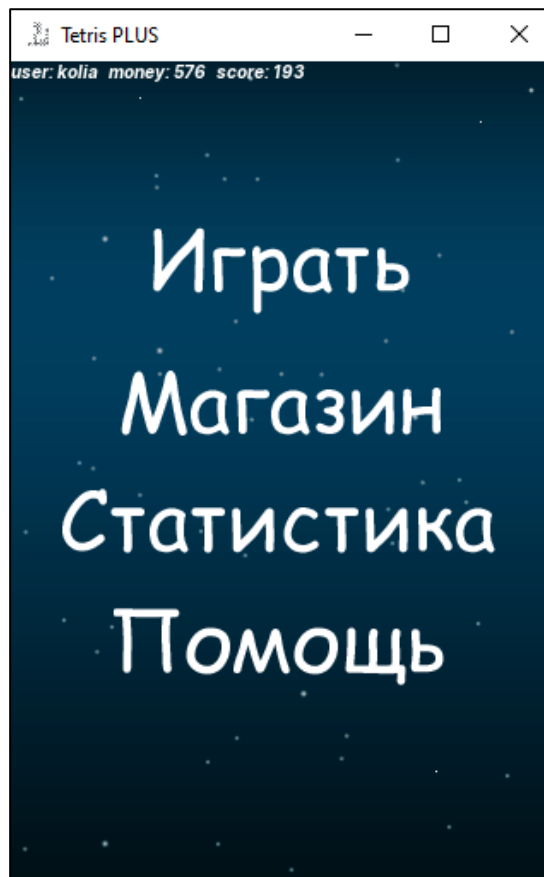


Рисунок 2.10. Главное меню игры

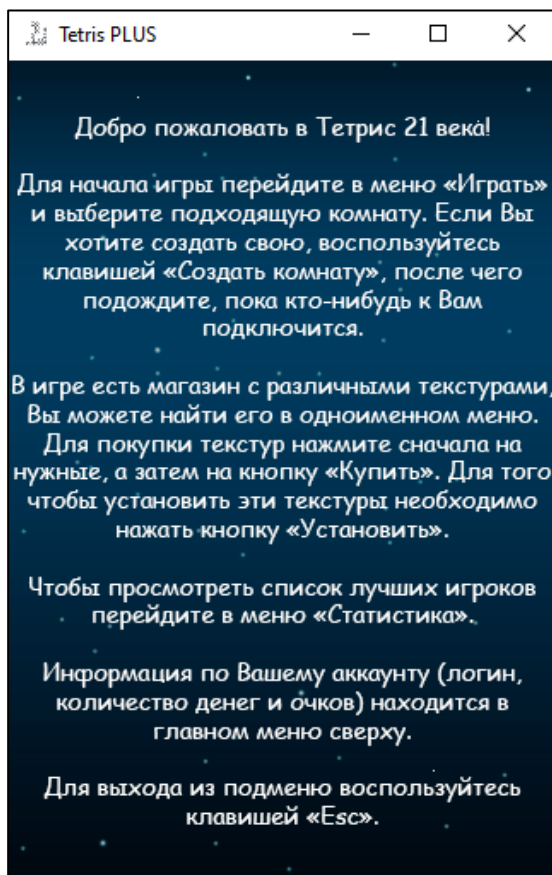
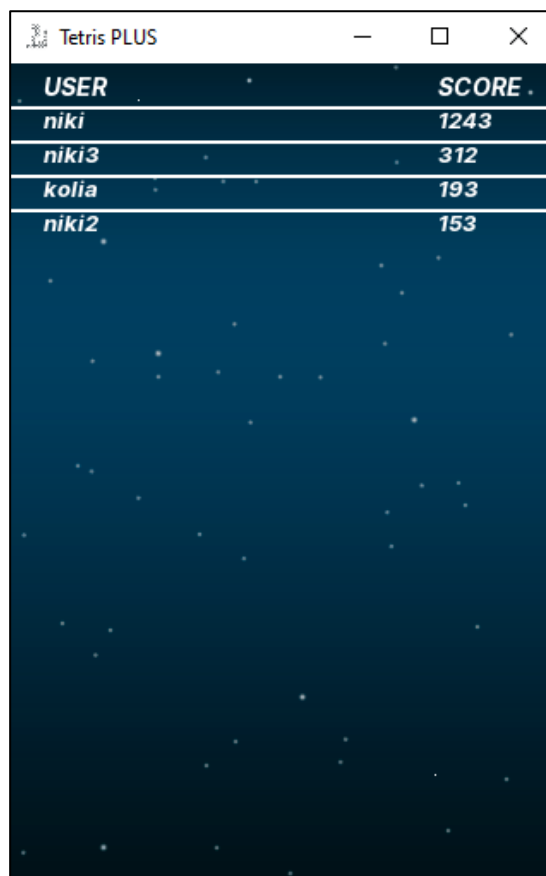


Рисунок 2.11. Подменю «Помощь».



The screenshot shows a window titled "Tetris PLUS" with a standard Windows-style title bar (minimize, maximize, close buttons). Below the title bar is a table with two columns: "USER" and "SCORE". The table lists four players: niki (1243), niki3 (312), kolia (193), and niki2 (153). The background of the window is a dark blue space with white stars.

USER	SCORE
niki	1243
niki3	312
kolia	193
niki2	153

Рисунок 2.12. Статистика игроков.

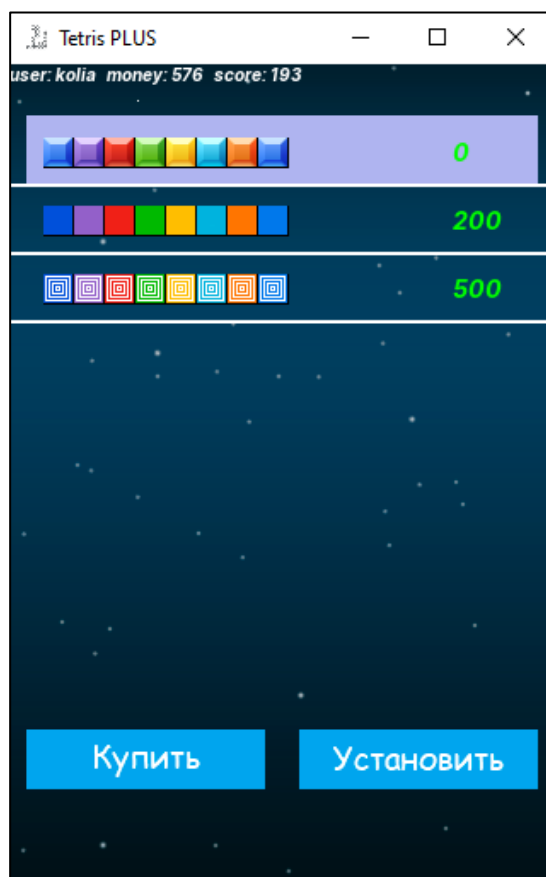


Рисунок 2.13. Подменю «Магазин».

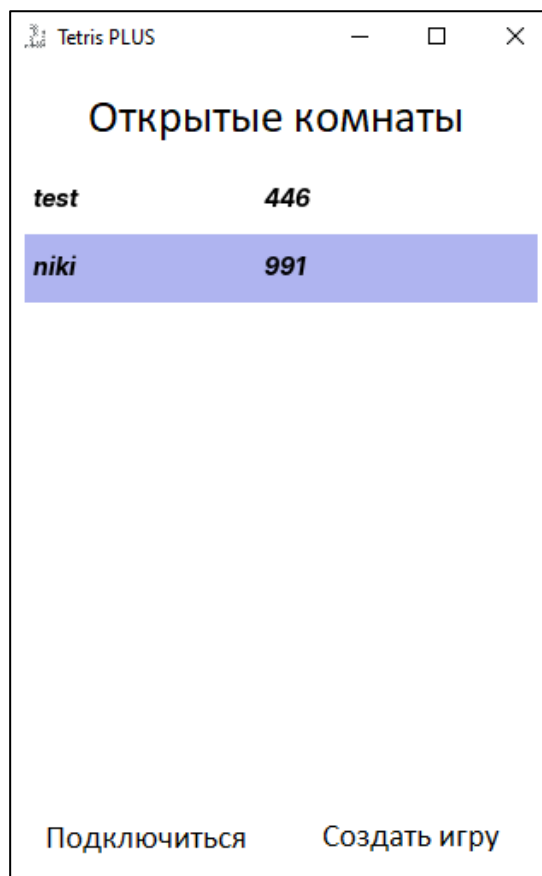


Рисунок 2.14. Подменю комнат для начала игры.

## ГЛАВА 3

### ТЕСТИРОВАНИЕ ПРОГРАММЫ

#### 3.1 Тестирование исходного кода

Тесты для проверки исходного кода проекта были предоставлены, однако само тестирование проведено не было.

#### 3.2 Тестирование интерфейса пользователя и юзабилити

Проект делится на два больших модуля: сервер и клиент. Все тесты на проверку данных модулей прошли успешно. Всего было четыре теста. Первый тест был направлен на работоспособность сервера и клиента, а также их корректное завершение работы (рисунок 3.1). Второй тест был направлен на проверку работоспособности главного меню клиента, модуля авторизации и регистрации (рисунок 3.2). Третий тест (рисунок 3.3) проверял реализацию комнат в подменю «Играть»: подключение к уже существующим и создание новой. Последний тест был направлен на проверку работоспособности игры (рисунок 3.4): перемещение фигур по игровому полю по таймеру и с помощью стрелок влево и вправо, вращение фигуры относительно своего центра с помощью стрелки вверх, собирание линии и начисление за это очков рейтинга и монет.

Тест 1/Test Name 1

Шаг / Step No.	Действие (операция) / Process (Actions)	Ожидаемый результат / Expected Results	Результат / Actual Results	Пройден / не пройден / не доступен*	Комментарии / Notes
1	1) Запуск сервера ("tetris-plus\Debug\ tetris-plus.exe). Ввод порта 1234 (другие порты будут поддерживаться с выходом новых обновлений) 2) Запуск двух клиентов (main.py.py)	Сервер запускается – открывается окно консоли. После запуска клиентов открываются окна для ввода логина и пароля или регистрации (на выбор), в консоли подтверждаются подключения и корректность входа клиентов или регистрации нового пользователя.	Соответствует ожидаемому результату.	Пройден.	
2	Нажатие на кнопку закрытия окна	Окно клиента закрывается, в консоли сервера указывается что данный клиент отключен.	Окно клиента закрывается, сервер пишет об отключении пользователя.	Пройден.	

Рисунок 3.1. Тест сервера.

## Тест 2/Test Name 2

Шаг /Step No.	Действие (операция) /Process (Actions)	Ожидаемый результат /Expected Results	Результат /Actual Results	Пройден /не пройден/ не доступен*	Комментарии /Notes
1	После входа зайти во все пункты меню. Для выхода использовать клавишу «Esc».	При входе в меню «Играть» должна появиться таблица открытых комнат, а также кнопка для создания собственной комнаты. При входе в меню «Статистика» должна отобразиться таблица топ 10 игроков и их рейтинг. При входе в меню «Магазин» должно отобразиться меню магазина: текстуры и их цены. При входе в меню «Помощь» должен отобразиться текст краткого описания действий для начала игры.	Соответствует ожидаемому результату.	Пройден.	

Рисунок 3.2. Тест клиента.

## Тест 3/Test Name 3

Шаг /Step No.	Действие (операция) /Process (Actions)	Ожидаемый результат /Expected Results	Результат /Actual Results	Пройден /не пройден/ не доступен*	Комментарии /Notes
1	Проверка реализации комнат, поиска игроков.	При входе в меню «Играть» должна появиться таблица открытых комнат, а также кнопка для создания собственной комнаты. При нажатии на кнопку «Создать комнату» должен появиться значок загрузки, как только кто-то выберет данную комнату, должна начаться игра. При нажатии на любую доступную комнату и кнопки «Подключиться» должна начаться игра.	Соответствует ожидаемому результату.	Пройден.	

Рисунок 3.3. Тест подключения к серверу при авторизации

## Тест 4/Test Name 4

Шаг /Step No.	Действие (операция) /Process (Actions)	Ожидаемый результат /Expected Results	Результат /Actual Results	Пройден /не пройден/ не доступен*	Комментарии /Notes
1	После начала игры необходимо управлять падающей фигурой при помощи стрелок влево и вправо для перемещения фигуры по горизонтали, и стрелкой вверх – для поворота фигуры.	В начале игры фигура начнет падать вниз, при нажатии стрелок фигура должна перемещаться в нужном направлении, а также выполнять поворот. Выйти за пределы игрового поля невозможно. При касании нижней части поля фигура замирает, управлять ей более нельзя, появляется новая фигура и начинается падение, ей управляет 2й игрок. При заполнении горизонтальной линии – она исчезает, а фигуры сверху спускаются вниз. Игра заканчивается при достижении верхнего края поля одной фигурой, побеждает игрок, собравший большее количество линий.	Соответствует ожидаемому результату.	Пройден.	Как таковой победы нет, каждый игрок получает определенное количество баллов и монет, в зависимости от собранных линий.

Рисунок 3.4. Тест для проверки работоспособности игрового меню



## ГЛАВА 4

### ПЕРСПЕКТИВЫ ДАЛЬНЕЙШЕГО РАЗВИТИЯ ПРОЕКТА

#### 4.1 Перспективы технического развития

Во время реализации данного проекта были реализованы все задачи поставленные в начале работы. Однако из-за недостатка опыта разработки клиент-серверных приложений проект получился прост, его внешний вид получился приемлемым, но не для современности. Библиотека SFML оказалась не лучшим решением для создания интерфейса игры. Ее возможностей не хватает для создания современной графики в играх.

Проект не имеет прямых аналогов, однако маловероятно, что он будет интересен пользователям в таком виде. В дальнейшем планируется переработка графической составляющей игры с использованием анимации. Также планируется добавление альтернативных режимов, дополнительных предметов в магазине и новых функций о время игры.

#### 4.2 Перспективы монетизации

На данном этапе монетизация не планируется, но после доработок игры, есть возможность подключения контекстной рекламы.

## ЗАКЛЮЧЕНИЕ

В результате работы над данным проектом был получен опыт в разработке проектов в среде разработки Microsoft Visual Studio и IDLE, опыт использования текстового редактора Sublime Text 3 и установки дополнительных модулей для него, распределении времени и выставления приоритетов в процессе разработки. Так же был получен опыт в поиске и выборе библиотек, позволяющих реализовать функционал, необходимый в приложении. Был получен опыт работы со сторонними, ранее не изученными библиотеками, а именно SFML, socket и румysql. Освоен принцип реализации работы клиент-серверных структур.

В процессе разработки были решены все поставленные на этапе формирования проекта задачи, что привело к достижению цели, а именно был реализован клиент-серверный многопользовательский вариант игры «Тетрис».

Проект, находится на ранних этапах разработки, и все еще нуждается в доработке, интерфейс программы очень простой, в некоторых местах все еще можно найти грубоватые заготовки или не полностью продуманные функции, однако весь функционал в программе рабочий и выполняет все задачи, поставленные в начале разработки.

**ЛИТЕРАТУРА**

1. Оформление выпускной квалификационной работы на соискание квалификационного уровня «Магистр» («Бакалавр»): методические рекомендации. / сост. Бержанский В.Н., Дзедолик И.В., Полулях С.Н. – Симферополь: КФУ им. В.И.Вернадского, 2017. – 31 с.
2. Пример клиента на Python и сервера на C++ с использованием сокетов [электронный ресурс] - режим доступа: <https://coderoad.ru/13466341/Пример-для-простого-клиента-Python-и-сервера-C>, дата обращения: 31.05.2021;
3. C++ использование сокетов [электронный ресурс] - режим доступа: [https://code-live.ru/post/cpp-http-server-over-sockets/#\\_2](https://code-live.ru/post/cpp-http-server-over-sockets/#_2), дата обращения: 31.05.2021;
4. Игра Тетрис на двоих [электронный ресурс] - режим доступа: <https://www.min2win.ru/gm.php?id=839>, дата обращения: 31.05.2021;
5. Игра Тетрис на двоих [электронный ресурс] - режим доступа: <http://playmap.ru/flash/tetris/7270-tetris-na-dvoih.html>, дата обращения: 31.05.2021;
6. Игра Tetris Cube [электронный ресурс] - режим доступа: <https://hoopgame.net/play/Tetris-Cube>, дата обращения: 31.05.2021;
7. Руководство конфигурации проекта для работы с библиотекой SFML в среде Microsoft Visual Studio «SFML and Visual Studio» [электронный ресурс] - режим доступа: <https://www.sfml-dev.org/tutorials/2.5/start-vc.php>, дата обращения: 31.05.2021;
8. Уроки по графической библиотеке SFML [электронный ресурс] - режим доступа: <https://ravesli.com/uroki-po-sfml/>, дата обращения: 31.05.2021;
9. Документация по языку C++ [электронный ресурс] - режим доступа: <https://ru.cppreference.com/w/>, дата обращения: 31.05.2021;