МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ»
Кафедра «Измерительно-вычислительные комплексы»

КУРСОВАЯ РАБОТА
по дисциплине «Алгоритмы и структуры данных»
Тема: «Компьютерная игра “Фризские шашки - Поддавки”»
Р.02069337. 23/738 14 ПЗ-01
Листов 28



Руководитель разработки:
к.т.н, доцент кафедры "ИВК"
Шишкин Вадим Викторинович
« » 2025 г.

Исполнитель:
студент гр. ИСТбд-22
Кокарев Ярослав Владимирович
« » 2025 г.

2025

Содержание

Аннотация

Проект представляет собой движок для игры в шашки, не предназначенный для игры против компьютера или другого игрока.

Движок предназначен для автоматизации процесса игры в шашки. Пользователи – студенты, обучающиеся программированию, желающие получить опыт в разработке игр на Python. Автоматизируемые процессы: отображение доски, перемещение шашек.

Игра разработана на языке программирования Python с использованием библиотеки Tkinter для создания графического интерфейса. Выбор реализации данной игры обусловлен её популярностью и простотой, что делает её подходящей для изучения основ программирования и разработки графических интерфейсов. Проект позволяет сосредоточиться на принципах работы с графикой, обработкой событий и взаимодействием с пользователем.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение
высшего образования
«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ»

Кафедра «Измерительно-вычислительные комплексы»

ТЕХНИЧЕСКОЕ ЗАДАНИЕ
по дисциплине «Алгоритмы и структуры данных»
Тема: «Компьютерная игра “Фризские шашки - поддавки”»
Р.02069337. 23/738 14 ПЗ-01
Листов 3

Инв. | Полп. и | Взам | Инв. | Полп. и

Руководитель разработки:
к.т.н, доцент кафедры "ИВК"
Шишкин Вадим Викторинович
«  » 2025 г.

Исполнитель:
студент гр. ИСТбд-22
Кокарев Ярослав Владимирович
«  » 2025 г.

2025

**Введение**

Наименование разрабатываемого приложения: "Фризские шашки - поддавки".
Данная программа представляет собой простой движок для игры в шашки, не предназначенный для игры против компьютера или другого игрока.

**1. Основания для разработки**

Учебная программа по дисциплине «Алгоритмы и структуры данных» по специальности 09.03.02 «Информационные системы и технологии».

**2. Требования к программе или программному изделию**

**2.1. Функциональное назначение**

Движок предназначен для автоматизации процесса игры в шашки. Пользователи – студенты, обучающиеся программированию, желающие получить опыт в разработке игр на Python.
Автоматизируемые процессы: отображение доски, перемещение шашек.

**2.2. Требования к функциональным характеристикам**

**2.2.1 Требования к структуре приложения**

Приложение должно иметь модульную организацию, состоящую из следующих компонентов:

Главный модуль (UI)

Модуль обработки логики игры

Модуль управления состоянием игры

### 2.2.2 Требования к составу функций приложения

Основные функции приложения:

Регистрация

Отображение игровой доски

Перемещение шашек между клетками

Проверка правил ходов

### 2.2.3 Требования к организации информационного обеспечения

Пользовательский интерфейс должен быть интуитивно понятным, с возможностью взаимодействия по клику мыши. Входные данные: координаты клеток для перемещения шашек. Выходные данные: обновленная игровая доска.

## 2.3. Требования к надежности

Приложение должно оставаться работоспособным при нестандартных вводах.

## 2.4. Требования к информационной и программной совместимости

Программа разработана на Python 3.13 и использует библиотеку tkinter для графического интерфейса. Операционная система – Windows.

## 2.5. Требования к маркировке и упаковке

Определяются заданием на курсовую работу.

### 2.6. Требования к транспортированию и хранению

#### 2.6.1 Условия транспортирования

Требования к условиям транспортирования не предъявляются.

#### 2.6.2 Условия хранения

Обеспечение свободного доступа к проекту в репозитории до окончания срока учебы.

#### 2.6.3 Сроки хранения

Срок хранения – до окончания срока учебы.

## 3. Требования к программной документации

Определяются заданием на курсовую работу.

## 4. Стадии и этапы разработки

Определяются заданием на курсовую работу.

## 5. Порядок контроля и приемки

Определяются заданием на курсовую работу.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ»
Кафедра «Измерительно-вычислительные комплексы»

# ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
по дисциплине «Алгоритмы и структуры данных»
Тема: «Компьютерная игра “Фризские шашки - Поддавки”»
Р.02069337. 23/738 14 ПЗ-01
Листов 4

Инв. | Полп. и | Взд | Инв | Полп. и

Руководитель разработки:
к.т.н, доцент кафедры "ИВК"
Шишкин Вадим Викторинович
« » 2025 г.

Исполнитель:
студент гр. ИСТбд-22
Кокарев Ярослав Владимирович
« » 2025 г.

2025

## Введение

В данной работе представлено приложение "Фризские шашки - поддавки", реализующее классическую настольную игру для двух игроков. Игра разработана на языке программирования Python с использованием библиотеки Tkinter для создания графического интерфейса. Приложение позволяет пользователям взаимодействовать с игровым полем, выбирать и перемещать шашки, а также проходит регистрацию перед началом игры.

Выбор реализации данной игры обусловлен её популярностью и простотой, что делает её подходящей для изучения основ программирования и разработки графических интерфейсов. Проект позволяет сосредоточиться на принципах работы с графикой, обработкой событий и взаимодействием с пользователем.

## Проектная часть

### 1.1. Постановка задачи на разработку приложения

Определяется заданием на курсовую работу. Детализируется в разработанном техническом задании.

### 1.2. Математические методы

В данной игре используется математическая модель, основанная на правилах шашек. Основные аспекты модели включают:

- Игровое поле: квадратная сетка размером 10x10, на которой располагаются шашки.
- Перемещение шашек: шашки могут двигаться по диагонали на одну клетку вперед или на две клетки с прыжком через шашку противника.
- Проверка ходов: реализована логика, проверяющая допустимость хода на основе текущего положения шашек и правил игры.

Выбор данной модели обоснован простотой правил и логики игры, что позволяет сосредоточиться на аспектах программирования и разработки интерфейса.

### 1.3. Архитектура и алгоритмы

#### 1.3.1. Архитектура

Архитектура приложения включает следующие основные структуры данных и функции:

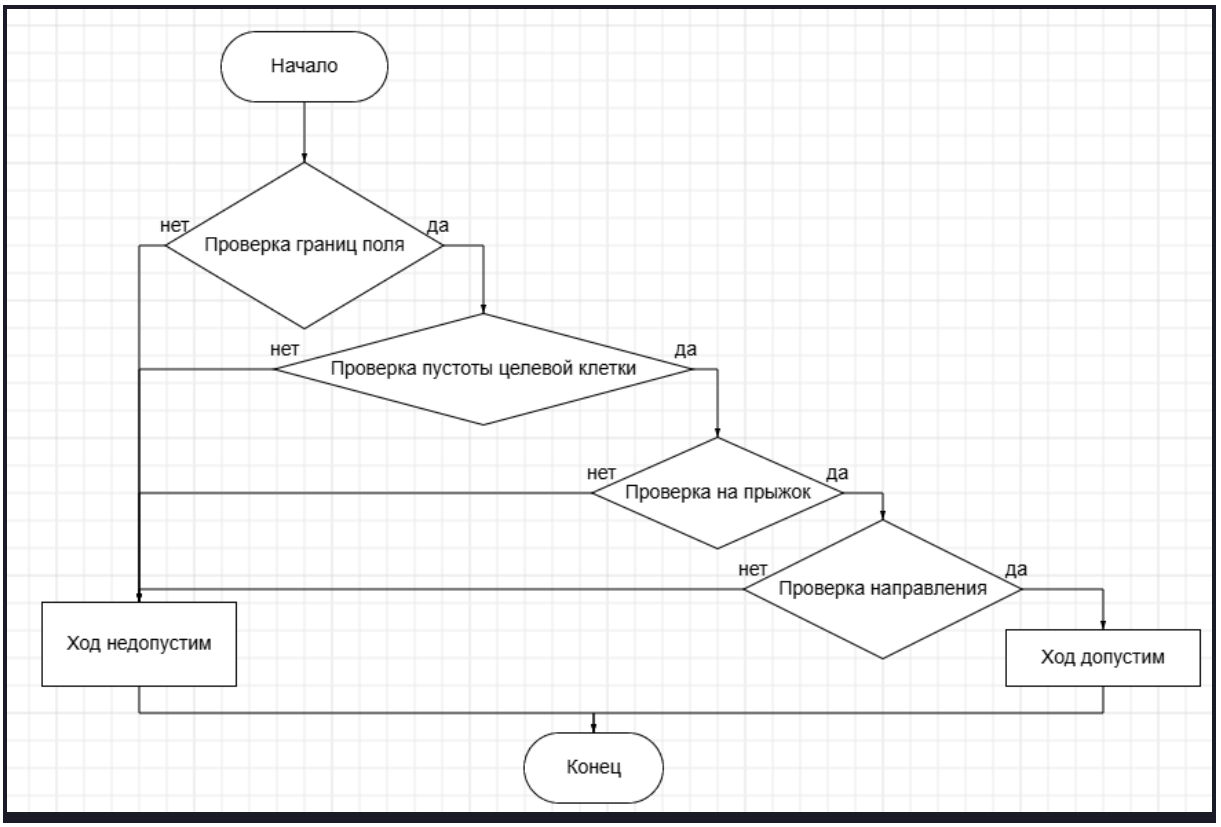
- Структура данных:
  - `board`: двумерный массив, представляющий игровое поле, где каждая ячейка может содержать информацию о шашке (цвет) или быть пустой.
  - `selected_piece`: переменная, хранящая информацию о выбранной шашке.
  - `current_player`: переменная, определяющая текущего игрока (черный или белый).
- Основные функции:
  - `create_board()`: инициализация игрового поля.
  - `draw_board()`: отрисовка игрового поля и шашек на экране.
  - `click_handler(event)`: обработка событий клика мыши, выбор и перемещение шашек.
  - `is_valid_move(row, col)`: проверка допустимости хода.
  - `select_piece(row, col)`: выбор шашки для перемещения.
  - `move_piece(row, col)`: выполнение перемещения шашки.
  - `show_registration()`: отображение окна регистрации.

### 1.3.2. Алгоритм проверки допустимости хода

Алгоритм проверки допустимости хода включает следующие шаги:

1. Проверка, находится ли целевая клетка в пределах игрового поля.
2. Проверка, является ли целевая клетка пустой.
3. Если шашка перемещается на две клетки, проверяется наличие шашки противника на промежуточной клетке.
4. Проверка, соответствует ли направление движения правилам для текущего игрока (только вперед для обычных шашек).

Блок-схема алгоритма проверки допустимости хода:

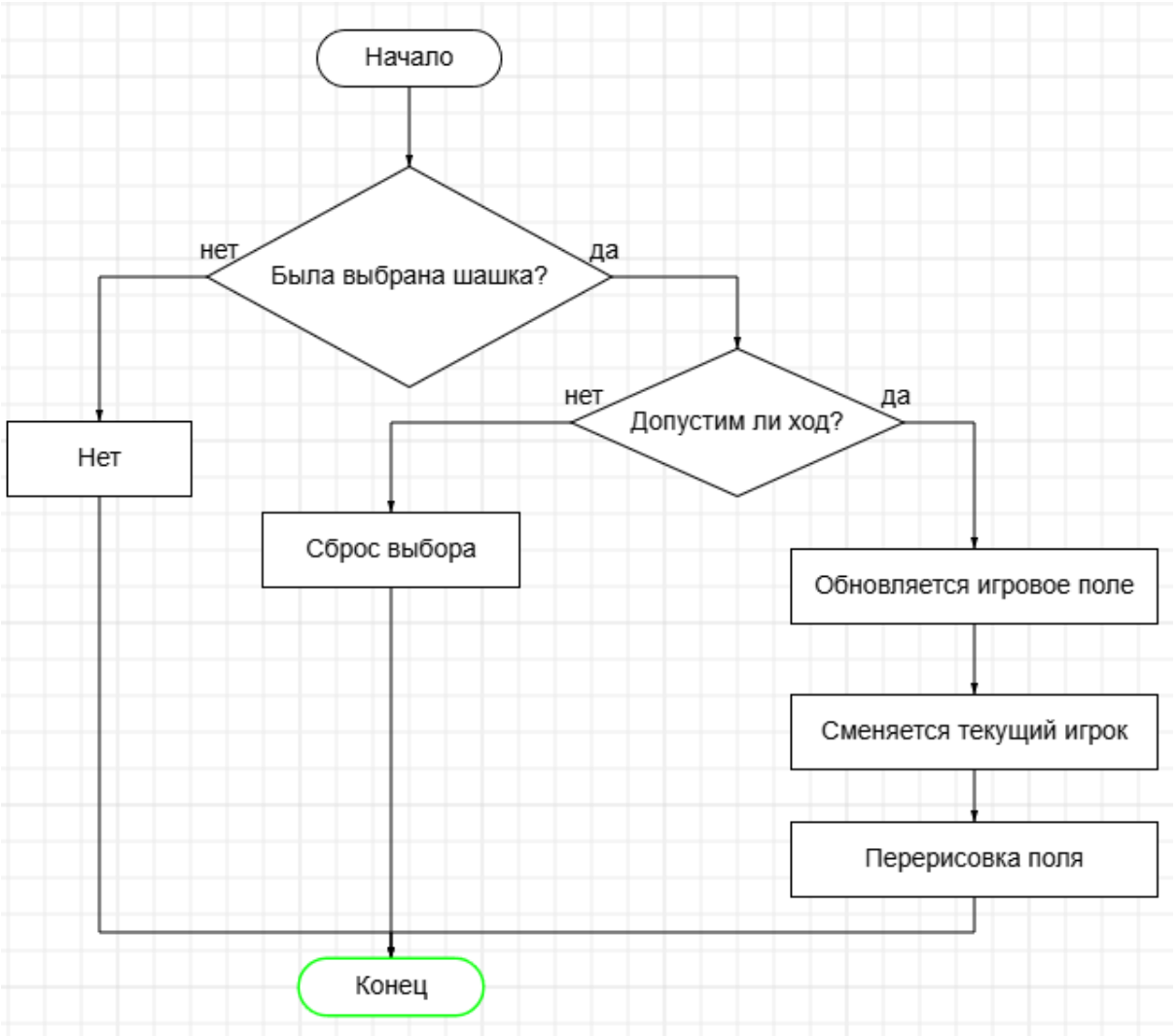


### 1.3.3. Алгоритм перемещения шашки

Алгоритм перемещения шашки включает следующие шаги:

1. Проверка, была ли выбрана шашка.
2. Проверка допустимости хода с помощью функции `is_valid_move()`.
3. Если ход допустим, перемещение шашки:
    - Обновление состояния игрового поля.
    - Смена текущего игрока.
4. Перерисовка игрового поля.

Блок-схема алгоритма перемещения шашки:



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ»
Кафедра «Измерительно-вычислительные комплексы»

Подп. и
Инв
Взз
Подп. и
Инв.

# РУКОВОДСТВО ПРОГРАММИСТА

по дисциплине «Алгоритмы и структуры данных»

Тема: «Компьютерная игра “Фризские шашки - Поддавки”»

Р.02069337. 23/738 14 ПЗ-01

Листов 4

Руководитель разработки:
к.т.н, доцент кафедры "ИВК"
Шишкин Вадим Викторинович
« » 2025 г.

Исполнитель:
студент гр. ИСТбд-22
Кокарев Ярослав Владимирович
« » 2025 г.

2025

## 1. Назначение и условия применения программы

### 1.1. Назначение и функции, выполняемые приложением

Приложение "Фризские шашки" предназначено для реализации классической настольной игры шашки в графическом интерфейсе. Игра поддерживает двух игроков, которые могут взаимодействовать с игровым полем, выбирать и перемещать шашки. Основные функции приложения включают:

- Создание игрового поля: Игра начинается с инициализации квадратной сетки 10x10, на которой располагаются шашки каждого игрока.
- Выбор и перемещение шашек: Игроки могут выбирать свои шашки и перемещать их по полю в соответствии с правилами игры.
- Проверка допустимости ходов: Логика игры включает проверку на корректность выбранного хода, включая возможность прыжка через шашки противника.
- Регистрация пользователей: Перед началом игры пользователи должны пройти процесс регистрации, что позволяет сохранить индивидуальные данные для каждого игрока.

Правила игры в шашки заключаются в том, что игроки по очереди перемещают свои шашки на одну клетку вперед по диагонали или могут перепрыгивать через шашки противника, если это возможно. Цель игры — захватить все шашки противника.

### 1.2. Условия, необходимые для использования приложения

Для корректной работы приложения "Фризские шашки" необходимы следующие условия:

- Операционная система: Приложение совместимо с операционными системами Windows, macOS и Linux.

- Платформа: Необходима установка Python версии 3.6 или выше.
- Инструментальная среда: Для разработки и запуска приложения может использоваться любая IDE, поддерживающая Python, например, PyCharm или Visual Studio Code.
- Библиотеки: Необходима установка библиотеки Tkinter, которая входит в стандартную библиотеку Python и не требует отдельной установки.

## 2. Характеристики программы

### 2.1. Характеристики приложения

Приложение состоит из 120 значимых строк программного кода, которые выполняют различные действия, связанные с логикой игры и графическим интерфейсом. Основные структурные элементы приложения включают:

- Двумерный массив: Для представления игрового поля используется массив размером 10x10, где каждая ячейка может содержать информацию о шашке или быть пустой.
- Переменные: Для хранения состояния игры, таких как текущий игрок и выбранная шашка.

Используемые библиотеки:

- `tkinter`: Библиотека для создания графического интерфейса.
- `simpledialog` и `messagebox`: Модули из Tkinter для создания диалоговых окон.

Работа приложения начинается с создания графического окна, после чего пользователи проходят регистрацию. Затем они могут начать игру, выбирая и перемещая шашки, при этом приложение контролирует корректность ходов.

### 2.2. Особенности реализации приложения

В приложении используются следующие структуры данных:

- Двумерный массив `board`: Этот массив представляет игровое поле. Каждая ячейка может содержать строку, указывающую цвет шашки ('black' или 'white'), или быть пустой строкой ('').
- Переменные `selected_piece` и `current_player`: `selected_piece` хранит координаты выбранной шашки, а `current_player` определяет, чей сейчас ход.

Выбор двумерного массива для представления игрового поля обусловлен его простотой и удобством для выполнения операций, таких как проверка допустимости ходов и перемещение шашек.

## 3. Обращение к программе

### 3.1. Методы и алгоритмы

Приложение включает следующие ключевые методы:

- `__init__(self, master)`: Инициализация игрового поля и графического интерфейса.
- `create_board(self)`: Создание и инициализация игрового поля.
- `draw_board(self)`: Отрисовка игрового поля и шашек на экране.
- `click_handler(self, event)`: Обработка кликов мыши для выбора и перемещения шашек.
- `select_piece(self, row, col)`: Выбор шашки для перемещения.
- `move_piece(self, row, col)`: Выполнение перемещения шашки.
- `is_valid_move(self, row, col)`: Проверка допустимости хода.
- `show_registration(self)`: Отображение окна регистрации.

### 3.2. Используемые библиотеки

- `tkinter`: Основная библиотека для создания графического интерфейса.
- `simpledialog`: Используется для создания диалоговых окон ввода.
- `messagebox`: Используется для отображения сообщений пользователю.

## 4. Сообщения

Приложение выдает следующие сообщения в результате контроля корректности ввода/вывода:

- При успешной регистрации: "Регистрация успешна!"
- При ошибке регистрации: "Логин и пароль не могут быть пустыми."
- При попытке сделать недопустимый ход: В программе не предусмотрено отображение сообщений для недопустимых ходов, однако логика игры не позволяет выполнить такие действия.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ»
Кафедра «Измерительно-вычислительные комплексы»

# ТЕСТОВАЯ ДОКУМЕНТАЦИЯ
по дисциплине «Алгоритмы и структуры данных»
Тема: «Компьютерная игра “Фризские шашки - Поддавки”»
Р.02069337. 23/738 14 ПЗ-01
Листов 5



Руководитель разработки:
к.т.н, доцент кафедры "ИВК"
Шишкин Вадим Викторинович
« » 2025 г.

Исполнитель:
студент гр. ИСТбд-22
Кокарев Ярослав Владимирович
« » 2025 г.

2025

| ID | Название | Предусловия | Шаги | Ожидаемый результат |
|---|---|---|---|---|
| TC 1 | Инициализация исходного состояния | Программа запущена | Запустить программу | Должно появиться шашечное поле 10x10 и экран регистрации. |
| TC 2 | Проверка регистрации | Программа запущена | Ввести логин и пароль и нажать "Зарегистрироваться" | Сообщение об успешной регистрации. |
| TC 3 | Проверка пустого логина | Программа запущена | Нажать "Зарегистрироваться" без ввода логина | Сообщение об ошибке: "Логин и пароль не могут быть пустыми." |
| TC 4 | Проверка пустого пароля | Программа запущена | Нажать "Зарегистрироваться" без ввода пароля | Сообщение об ошибке: "Логин и пароль не могут быть пустыми." |
| TC 5 | Проверка выбора шашки | Игровое поле | Нажать на шашку игрока | Шашка должна быть выделена. |

| | | | | |
|---|---|---|---|---|
| | | отображается | | |



| | | | | |
|---|---|---|---|---|
| ТС 6 | Проверка перемещения шашки | Шашка выбрана | Нажать на допустимую клетку | Шашка должна переместиться на выбранную клетку. |
| ТС 7 | Проверка недопустимого перемещения | Шашка выбрана | Нажать на недопустимую клетку | Шашка не должна переместиться. |
| ТС 8 | Проверка прыжка через шашку противника | Шашка выбрана | Нажать на клетку, через которую есть шашка противника | Шашка должна переместиться. |
| ТС 9 | Проверка смены игрока | Шашка перемещена | Проверить текущего игрока | Игрок должен смениться на противоположного. |
| ТС 10 | Проверка выхода из окна | Окно регистрации открыто | Закрыть окно регистрации | Окно регистрации |

| | | | | |
|---|---|---|---|---|
| | регистрации | | 20 | должно закрыться. |
| ТС 11 | Проверка отображения шашек | Игровое поле отображается | Проверить наличие шашек на поле | Должны быть шашки в правильных начальных позициях. |
| ТС 12 | Проверка клика по пустой клетке | Игровое поле отображается | Нажать на пустую клетку | Ничего не должно произойти. |
| ТС 13 | Проверка клика по шашке противника | Игровое поле отображается | Нажать на шашку противника | Ничего не должно произойти. |
| ТС 14 | Проверка регистрации без логина и пароля | Программа запущена | Нажать "Зарегистрироваться" без ввода | Сообщение об ошибке: "Логин и пароль не могут быть пустыми." |
| ТС 15 | Проверка перерисовки игрового поля | Игра идет | Переместить шашку | Игровое поле должно обновиться и отобразить новое состояние. |

| | | | | |
|---|---|---|---|---|
| ТС 16 | Проверка регистрации с длинным логином | Программа запущена | Ввести длинный логин и пароль | Сообщение об успешной регистрации. |
| ТС 17 | Проверка регистрации с длинным паролем | Программа запущена | Ввести логин и длинный пароль | Сообщение об успешной регистрации. |
| ТС 18 | Проверка возврата к главному окну после регистрации | Окно регистрации открыто | Нажать "Зарегистрироваться" | Окно регистрации должно закрыться, игра должна начаться. |
| ТС 19 | Проверка отрисовки доски | Программа запущена | Запустить программу | Доска должна отобразитьс я с чередующимися клетками. |
| ТС 20 | Проверка возможности игры после | Игрок зарегистрирован | Начать игру | Игра должна начаться без ошибок. |

| | | | | |
|---|---|---|---|---|
| | регистрации | 22 | | |
| ТС 21 | Проверка закрытия приложения | Программа запущена | Закрыть приложение | Приложение должно закрыться. |

Код программы

```
import tkinter as tk
from tkinter import simpledialog, messagebox

class CheckersGame:
    def __init__(self, master):
        self.master = master
        self.master.title("Фризские шашки")
        self.board_size = 10
        self.cell_size = 60
        self.canvas = tk.Canvas(master, width=self.board_size *
self.cell_size, height=self.board_size * self.cell_size)
        self.canvas.pack()
        self.center_window()
        self.board = self.create_board()
        self.selected_piece = None
        self.current_player = 'white'
        self.draw_board()
        self.canvas.bind("<Button-1>", self.click_handler)

        self.is_registered = False
        self.show_registration()

    def create_board(self):
        board = [[" for _ in range(self.board_size)] for _ in
range(self.board_size)]
        for row in range(self.board_size):
            for col in range(self.board_size):
                if (row + col) % 2 == 1:
                    if row < 4:
                        board[row][col] = 'black'
```

```
                elif row > 5:
                    board[row][col] = 'white'
        return board
```



```
    def draw_board(self):
        self.canvas.delete("all")
        for row in range(self.board_size):
            for col in range(self.board_size):
                color = 'white' if (row + col) % 2 == 0 else 'gray'
                self.canvas.create_rectangle(col * self.cell_size, row *
self.cell_size,
                                             (col + 1) * self.cell_size, (row + 1) *
self.cell_size,
                                             fill=color)
                piece = self.board[row][col]
                if piece:
                    fill_color = 'black' if piece.startswith('black') else 'white'
                    self.canvas.create_oval(col * self.cell_size + 10, row *
self.cell_size + 10,
                                            (col + 1) * self.cell_size - 10, (row + 1) *
self.cell_size - 10,
                                            fill=fill_color)


    def center_window(self):
        width = self.master.winfo_screenwidth()
        height = self.master.winfo_screenheight()
        x = (width // 2) - (self.board_size * self.cell_size // 2)
        y = (height // 2) - (self.board_size * self.cell_size // 2)
        self.master.geometry(f'{self.board_size *
self.cell_size}x{self.board_size * self.cell_size}+{x}+{y}')

    def click_handler(self, event):
        if not self.is_registered:
            return
```

```
        col = event.x // self.cell_size
        row = event.y // self.cell_size
```



```
 if self.selected_piece:
            if self.is_valid_move(row, col):
                self.move_piece(row, col)
            else:
                self.selected_piece = None
                self.select_piece(row, col)
        else:
            self.select_piece(row, col)

    def select_piece(self, row, col):
        piece = self.board[row][col]
        if piece and (piece.startswith(self.current_player)):
            self.selected_piece = (row, col)
            self.draw_board()

    def move_piece(self, row, col):
        old_row, old_col = self.selected_piece
        self.board[row][col] = self.board[old_row][old_col]
        self.board[old_row][old_col] = ''
        self.selected_piece = None
        self.current_player = 'black' if self.current_player == 'white' else
'white'
        self.draw_board()

    def is_valid_move(self, row, col):
        if not (0 <= row < self.board_size and 0 <= col < self.board_size):
            return False

        piece = self.board[self.selected_piece[0]][self.selected_piece[1]]

        if self.board[row][col] != '':
            return False
```

```
        old_row, old_col = self.selected_piece
        jump_row = (old_row + row) // 2
        jump_col = (old_col + col) // 2
        if self.board[jump_row][jump_col] != '' and
self.board[jump_row][jump_col] != piece:
            return (abs(row - old_row) == 2) and (abs(col - old_col) == 2)

        if piece.startswith('white'):
            return (row - old_row == -1) and (abs(col - old_col) == 1)
        elif piece.startswith('black'):
            return (row - old_row == 1) and (abs(col - old_col) == 1)

    def show_registration(self):
        registration_window = tk.Toplevel(self.master)
        registration_window.title("Регистрация")
        registration_window.geometry("300x180")
        registration_window.attributes('-topmost', True)

        width = 300
        height = 180
        x = (self.master.winfo_screenwidth() // 2) - (width // 2)
        y = (self.master.winfo_screenheight() // 2) - (height // 2)
        registration_window.geometry(f"{width}x{height}+{x}+{y}")

        tk.Label(registration_window, text="Введите логин:").pack(pady=5)
        login_entry = tk.Entry(registration_window)
        login_entry.pack(pady=5)

        tk.Label(registration_window, text="Введите
пароль:").pack(pady=5)
        password_entry = tk.Entry(registration_window, show='*')
        password_entry.pack(pady=5)
```

```
        def on_register():
            login = login_entry.get()
            password = password_entry.get()
            if login and password:
                registration_window.withdraw()
                messagebox.showinfo("Регистрация", "Регистрация
успешна!", parent=self.master)
                self.is_registered = True  # Устанавливаем флаг регистрации
                registration_window.destroy()
            else:
                messagebox.showerror("Ошибка", "Логин и пароль не могут
быть пустыми.", parent=registration_window)

        tk.Button(registration_window, text="Зарегистрироваться",
command=on_register).pack(pady=10)

    def run(self):
        self.draw_board()
        self.master.mainloop()

if __name__ == "__main__":
    root = tk.Tk()
    game = CheckersGame(root)
    game.run()
```