МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Измерительно-вычислительные комплексы»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

на курсовую работу

по дисциплине «Алгоритмы и структуры данных»

Тема Компьютерная игра эндшпиль “Король, ладья - король, ладья” 

Р.02069337.22/2365-55 ТЗ-01

Листов 39

Руководитель разработки:

*Шишкин Вадим Викторинович*

« » 2023 г.

Исполнитель:

студент гр. ИСТбд-21

*Радченко Юлия Дмитриевна*

« » 2023 г.

2023

Содержание

Аннотация……...…………………………………………………...…...3

Техническое задание………….……………………………………….4

Пояснительная записка...……………………………………..……...9

Руководство программиста………………...………….……….…...17

Текст программы…..…………………………………………..……...28

**АННОТАЦИЯ**

Тема курсового проекта: Компьютерная игра эндшпиль “Король, ладья - король, ладья”.

Исполнитель:студент гр. ИСТбд-21 Радченко Юлия Дмитриевна

Руководитель разработки: Шишкин Вадим Викторинович

Работа состоит из технического задания, пояснительной записки, руководства программиста и текста программы.

В техническом задании описаны общие правила компьютерной игры эндшпиль “Король, ладья - король, ладья”, условия выигрыша и проигрыша, начальные позиции фигур. Приведены основания для разработки, функциональное назначение, основные требования к функциональным характеристикам, надежности, информационной и программной совместимости, хранению, транспортировке, программной документации.

В пояснительной записке указываются задачи, математические методы, архитектура и алгоритмы, тестирование и источники, использованные при разработке.

В руководстве программиста приводятся назначение и функции, выполняемые приложением, условия использования, характеристики приложения, особенности реализации приложения, обращение к программе, сообщения, выдаваемые по результатам контроля корректности ввода/вывода

Текст программы представляет собой полный код программы, реализующий компьютерную игру эндшпиль “Король, ладья - король, ладья”.

Ключевые слова: компьютерная игра, эндшпиль, шахматы, python.

Общий объём работы составляет 39 страниц.

*Приложение 2*

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Измерительно-вычислительные комплексы»

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

на курсовую работу

по дисциплине «Алгоритмы и структуры данных»

Тема Компьютерная игра эндшпиль “Король, ладья - король, ладья” 

Р.02069337.22/2365-55 ТЗ-01

Листов 5

Исполнитель:

студент гр. ИСТбд-21

*Радченко Юлия Дмитриевна*

« » 2023 г.

2023

**Введение**

**1. Наименование программы**

Однопользовательское десктопное приложение по игре эндшпиль «Король, ладья - Король, ладья» с графическим интерфейсом в среде Windows.

**2. Начальная позиция**

Игра ведется между двумя соперниками (пользователь - компьютер) шахматными фигурами разного цвета на шахматном квадратном поле размером 8х8 клеток. В начальной позиции у каждого из игроков по 2 шахматные фигуры - король и ладья. Начальная позиция выставляется случайным образом.

**3. Ходы игроков**

Игроки совершают ходы поочередно. Первый ход делают “белые”. В процессе хода игрок может переместить только одну шахматную фигуру своего цвета. Ладья ходит по вертикалям и горизонталям шахматной доски на любое расстояние и по всем направлениям: вверх, вниз, влево и вправо. Препятствием на ее движении являются фигуры: свои и противника. Ладья не может «перешагнуть» через свою фигуру или пешку, но может «взять» фигуру или пешку противника, заняв ее место. Король ходит и бьет во все стороны по вертикали, горизонтали и диагонали, но только на одно поле. Королю нельзя ходить на атакованное фигурой или пешкой противника поле, то есть под шах. Королю нельзя брать фигуру или пешку противника, находящуюся под защитой Расстояние между королями должно быть не меньше одного поля.

**4. Условия выигрыша и проигрыша**

Цель игры состоит в том, чтобы поставить королю противника мат – так напасть на него своей фигурой, чтобы противник не мог ни отступить королем, ни побить напавшую на него фигуру, ни защититься иным способом, перекрыв нападающей фигуре диагональ, вертикаль или горизонталь. Игра также прекращается в случае, когда игрок не может сделать ни одного реального хода, хотя его король не находится под шахом. Игрок, установивший мат, считается победителем, второй игрок – проигравшим. Если обе фигуры короля игроков не находятся под шахом, но невозможно сделать ход какой-либо фигурой так, чтобы не подставить короля под удар, – объявляется ничья.

**1. Основания для разработки**

Учебный план направления 09.03.02 «Информационные системы и технологии» (уровень образования: Высшее (Бакалавриат), форма обучения: Очная, срок обучения: 2022 - 2026) и распоряжение №12-р от 20.10.2023 о закреплении тем курсовых работ (проектов) за студентами 2 курса ФИСТ направления 09.03.02 “Информационные системы и технологии” (профиль Информационные системы и технологии) по дисциплине “Алгоритмы и структуры данных”

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

**2.1. Функциональное назначение**

Требуется разработать однопользовательское десктопное приложение по игре эндшпиль «Король, ладья - Король, ладья» с графическим интерфейсом в среде Windows.

**2.2 Требования к функциональным характеристикам**

2.2.1 Требования к структуре приложения

Приложение должно быть разработано в виде одного модуля с дополнительными информационными файлами при необходимости.

2.2.2 Требования к составу функций приложения

В приложении должны быть реализованы в графическом режиме следующие основные функции:

- регистрация/авторизация пользователя;

- отрисовка игрового поля;

- взаимодействие с пользователем;

- интерактивные приемы, проверка правильности и отрисовка хода пользователя;

- проверка окончания игры;

- вычисление, проверка правильности и отрисовка хода компьютера;

- информирование пользователя об окончании игры и победителе.

2.2.3 Требования к организации информационного обеспечения, входных и выходных данных

В приложении должен быть реализован графический интерфейс взаимодействия с пользователем. Изображения шахматных фигур могут храниться в отдельных графических файлах. Логин и пароль пользователя должны вводиться с клавиатуры. Логины и пароли зарегистрированных пользователей должны храниться в отдельном файле или базе данных в зашифрованном виде. Пояснительные информационные сообщения для пользователя должны выводиться внизу игрового поля по ходу игры.

**2.3 Требования к надёжности**

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением совокупности организационно-технических мероприятий, перечень которых приведен ниже:

1. организацией бесперебойного питания технических средств;

2. выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. «Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;

3. выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов;

4. необходимым уровнем квалификации сотрудников профильных подразделений.

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать времени, необходимого на перезагрузку операционной системы и запуск программы, при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Отказы программы возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой. Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу конечного пользователя без предоставления ему административных привилегий.

**2.4 Требования к информационной и программной совместимости**

Версия операционной системы: Windows 7, 8, 8.1, 10, 11

Язык программирования: Python

Среда разработки: PyCharm Community Edition 2023.1.1

Используемые библиотеки:

* графическая библиотека Tk
* библиотека PIL для работы с растровой графикой

**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

В электронном виде на сайте GitHub.

2.6.3 Сроки хранения

Срок хранения – до июля 2026 года.

**3. Требования к программной документации**

Определяются заданием на курсовую работу.

**4. Стадии и этапы разработки**

Определяются заданием на курсовую работу.

**5. Порядок контроля и приёмки**

Определяются заданием на курсовую работу.

*Приложение 2*

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра «Измерительно-вычислительные комплексы»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

на курсовую работу

по дисциплине «Алгоритмы и структуры данных»

Тема Компьютерная игра эндшпиль “Король, ладья - король, ладья”

Р.02069337.22/2365-55 ТЗ-01

Листов 8

Исполнитель:

студент гр. ИСТбд-21

*Радченко Юлия Дмитриевна*

« » 2023

2023

**Введение**

Компьютерная игра эндшпиль “Король, ладья - король, ладья”. Приложение разработано в виде одного модуля с дополнительными информационными файлами.

Основные функции приложения:

- регистрация/авторизация пользователя;

- отрисовка игрового поля;

- взаимодействие с пользователем;

- интерактивные приемы, проверка правильности и отрисовка хода пользователя;

- проверка окончания игры;

- вычисление, проверка правильности и отрисовка хода компьютера;

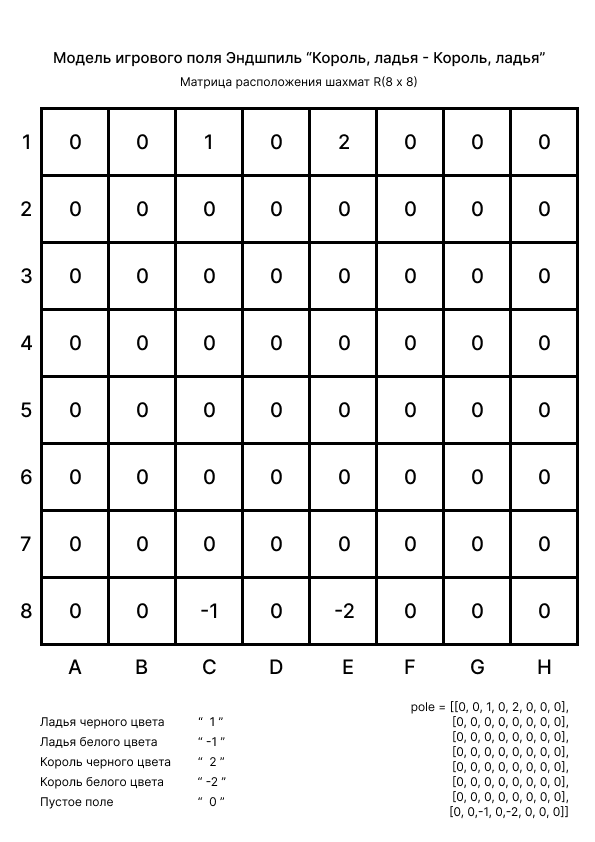
- информирование пользователя об окончании игры и победителе.

**1. Проектная часть**

**1.1 Постановка задачи на разработку приложения**

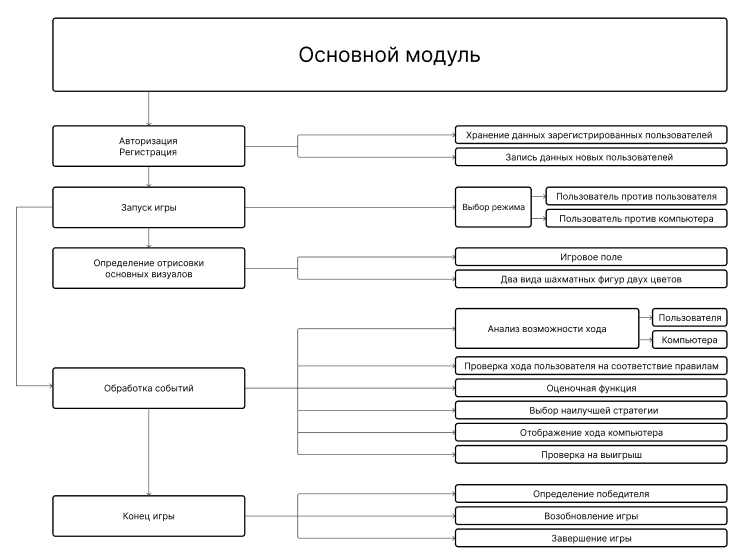
Определяется заданием на курсовую работу. Детализируется в разработанном техническом задании (приложение 1).

**1.2 Математические методы**



**1.3 Архитектура и алгоритмы**

1.3.1. Архитектура



1.3.2. Алгоритм проверки доступности хода

Для короля: если для короля выбран ход на соседние клетки прямо или наискосок, ход доступен.

Для ладьи: если для ладьи выбран ход на любое количество клеток по горизонтали или вертикали, учитывая, что другие фигуры не препятствуют этому, ход доступен.

1.3.3 Алгоритм хода пользователя

Первым нажатием на выбранную белую фигуру запускаются подсказки для хода королём или ладьёй в соответствии с правилами для данных фигур. Вторым нажатием осуществляется перемещение фигуры на выбранную клетку, если ход доступен. Иначе возникает сообщение о неверном ходе.

1.3.4 Алгоритм хода бота

После хода пользователя осуществляется запоминание позиций фигур на поле через переменные. Проверяется несколько условий и выполняется ход бота:

1. Если две ладьи на одной прямой, белая ладья съедается черной ладьей.
2. Если есть шах для черного короля, черный король перемещается наискосок, применяя Минимакс
3. Если черная ладья съедена, смещается король
4. Если нет шаха для черного короля и другие условия не подходят, смещается ладья

1.3.5 Алгоритм определения результата игры  
 1. Если белая и черная ладьи съедены, объявляется ничья.

2. Если количество шахов для белого короля достигло 10, объявляется победа черных.

3. Если количество шахов для черного короля достигло 10, объявляется победа белых.

**1.4. Тестирование**

Приводится интеллектуальная карта приложения.

1.4.1 Описание отчета о тестировании

В данном отчете представлены результаты тестирования программы на основе разработанных test-case и чек-листов, и статического тестирования документации и программного кода. Описаны проведенные тесты, их результаты и обнаруженные дефекты.

1.4.2 Цель тестирования

Цель тестирования: выявить ошибки, недоработки и несоответствия требованиям, а также оценить уровень качества ПО перед его выпуском. По результатам тестирования следует исправление выявленных багов.

1.4.3 Методика тестирования

Тестирование проводилось с использованием следующих методов:

1. Статическое тестирование: анализ и проверка кода без его запуска, выявление ошибок в технической документации.

2. Ручное тестирование: запуск пользовательских сценариев программы с различными входными данными и проверка корректности полученных результатов.

**1.4.4 Проведенные тесты**

В ходе тестирования были проведены следующие тесты:

**4.1)** Статическое тестирование

Количество обнаруженных и исправленных ошибок в документации: 11.

Количество обнаруженных и исправленных ошибок в программном коде: 726.

**4.2)** Ручное тестирование.

Написаны и проведены следующие тест-кейсы и чек-листы:

**ТК1. Отработка авторизации.**

Предварительные шаги:

Зарегистрироваться с логином TestTest и паролем 12345678.

Шаги:

1. Запустить приложение.

2. В окне регистрации, в поле «логин» ввести TestTest, а в поле «пароль» 12345678.

3. Нажать кнопку «Авторизоваться».

Ожидаемый результат: Пользователь начнет игру.

Фактический результат: Пользователь начал новую игру.

**ТК2. Прохождение регистрации** **односимвольными логином и паролем.**

Шаги:

1. Запустить приложение.

2. В окне регистрации, в поле «логин» ввести 1 и в поле «пароль» 1.

3. Нажать на кнопку «Зарегистрироваться/ начать игру».

Ожидаемый результат: Пользователь получит сообщение об ошибке.

Фактический результат: Пользователь получил сообщение об ошибке.

**ТК3. Прохождение регистрации** **с пустыми полями логина и пароля.**

Шаги:

1. Запустить приложение.

2. Нажать на кнопку «Зарегистрироваться/ начать игру».

Ожидаемый результат: Пользователь получит сообщение об ошибке.

Фактический результат: Пользователь получил сообщение об ошибке.

**ТК4. Прохождение регистрации** **с пробелами внутри логина и пароля.**

Шаги:

1. Запустить приложение.

2. В окне регистрации, в поле «логин» ввести Test Test, а в поле «пароль» 111 111.

3. Нажать на кнопку «Зарегистрироваться/ начать игру».

Ожидаемый результат: Пользователь получит сообщение об ошибке.

Фактический результат: Пользователь получил сообщение об ошибке.

Чек-лист формы регистрации

| Описание | Пример | Результат |
| --- | --- | --- |
| Поле «логин» | | |
| Корректный логин | TestTest | Сообщение об успешной регистрации |
| Пробелы внутри логина | Test Test | Сообщение об ошибке |
| Односимвольный логин | 1 | Сообщение об ошибке |
| Пустое поле логина |  | Сообщение об ошибке |
| Поле «пароль» | | |
| Корректный пароль | 12345678 | Сообщение об успешной регистрации |
| Пробелы внутри пароля | 111 111 | Сообщение об ошибке |
| Односимвольный пароль | 1 | Сообщение об ошибке |
| Пустое поле пароля |  | Сообщение об ошибке |

1.4.5 Выводы

На основе проведенных тестов сделаны следующие выводы:

1. Программа успешно прошла все тесты и работает корректно.

**2. Источники, использованные при разработке**

1. База данных шахматных окончаний [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85_%D1%88%D0%B0%D1%85%D0%BC%D0%B0%D1%82%D0%BD%D1%8B%D1%85_%D0%BE%D0%BA%D0%BE%D0%BD%D1%87%D0%B0%D0%BD%D0%B8%D0%B9>
2. Документация библиотеки tkinter [Электронный ресурс]. – Режим доступа: <https://tkdocs.com/index.html>
3. Ладейные окончания [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D0%B4%D0%B5%D0%B9%D0%BD%D1%8B%D0%B5_%D0%BE%D0%BA%D0%BE%D0%BD%D1%87%D0%B0%D0%BD%D0%B8%D1%8F>
4. Шишкин, В.В. Разработка логических компьютерных игр с графическим интерфейсом в среде питон / В.В. Шишкин, Д.С. Афонин. - Ульяновск: УлГТУ, 2023. – 89 с. – URL: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/<http://lib.ulstu.ru/venec/disk/2023/112.pdf>. – Текст: электронный.
5. Эндшпиль [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%AD%D0%BD%D0%B4%D1%88%D0%BF%D0%B8%D0%BB%D1%8C

*Приложение 3*

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра «Измерительно-вычислительные комплексы»

**РУКОВОДСТВО ПРОГРАММИСТА**

на курсовую работу

по дисциплине «Алгоритмы и структуры данных»

ТемаКомпьютерная игра эндшпиль “Король, ладья - король, ладья” 

Р.02069337.22/2365-55 ТЗ-01

Листов 11

Исполнитель:

студент гр. ИСТбд-21

*Радченко Юлия Дмитриевна*

« » 2023

2023

**1. Назначение и условия применения программы**

**1.1 Назначение и функции, выполняемые приложением**

Однопользовательское десктопное приложение по игре эндшпиль «Король, ладья - Король, ладья» с графическим интерфейсом в среде Windows.

**1.2 Условия, необходимые для использования приложения**

Версия операционной системы: Windows 7, 8, 8.1, 10, 11

Язык программирования: Python

Среда разработки: PyCharm Community Edition 2023.1.1

Используемые библиотеки:

* графическая библиотека Tk
* библиотека PIL для работы с растровой графикой

**2. Характеристики программы**

**2.1 Характеристики приложения**

Количество значимых строк программного кода: 544.

Количество структур данных: 3.

Количество алгоритмов: 4.

Используемые библиотеки:

* графическая библиотека Tk
* библиотека PIL для работы с растровой графикой

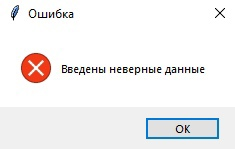
Работа приложения:

1. Окно регистрации и авторизации

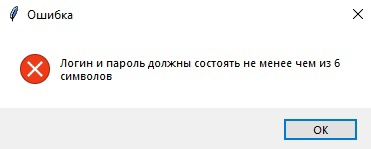
1.1. Для начала игры необходимо зарегистрироваться и авторизоваться



1.2. Если никаких данных введено не было или были введены пробелы, возникнет сообщение об ошибке:

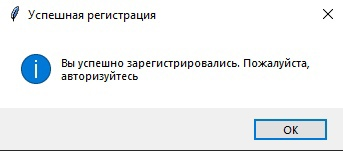


1.3. Если введенные при регистрации пароль и логин меньше шести символов, возникнет сообщение об ошибке:

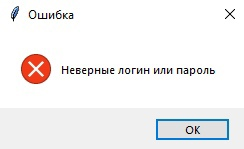


1.4. При введении корректных данных возникнет сообщение об успешной регистрации



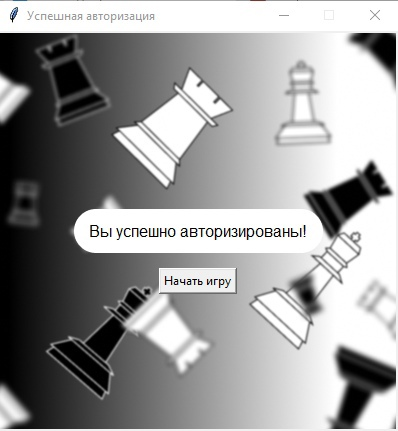


1.5. При вводе неправильного логина или пароля возникнет сообщение об ошибке:



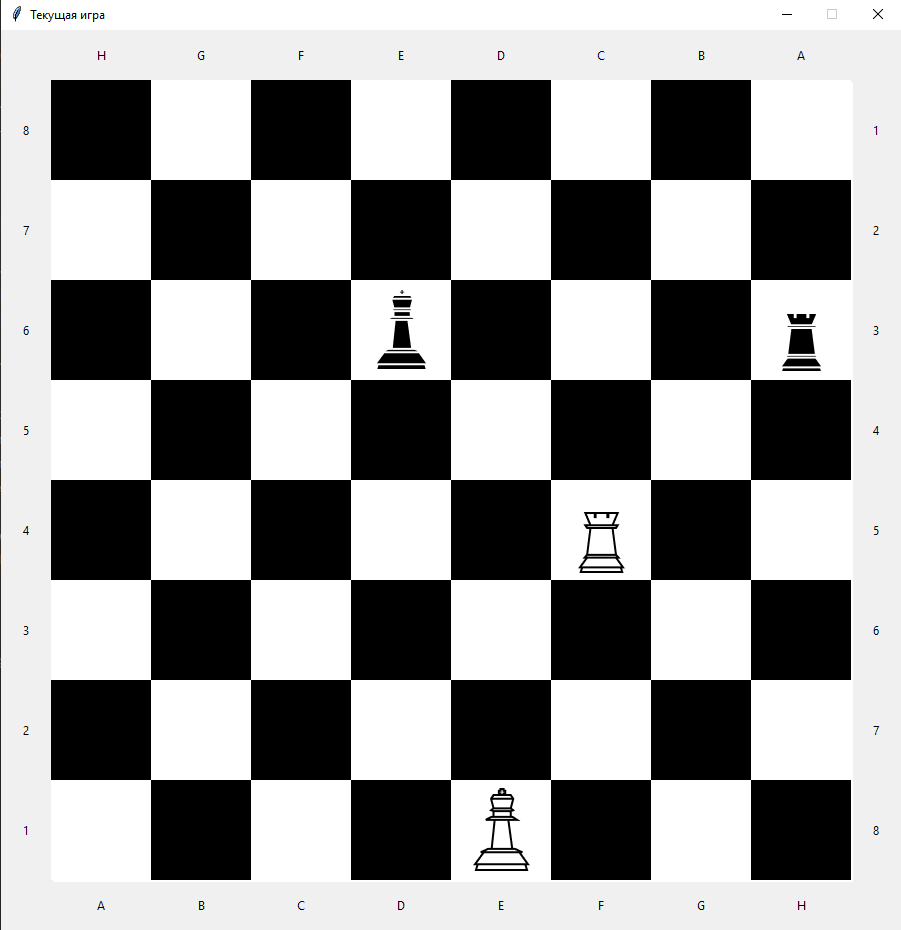
1.6. При вводе корректных данных появится новое окно с сообщением об успешной авторизации и кнопкой начала игры:

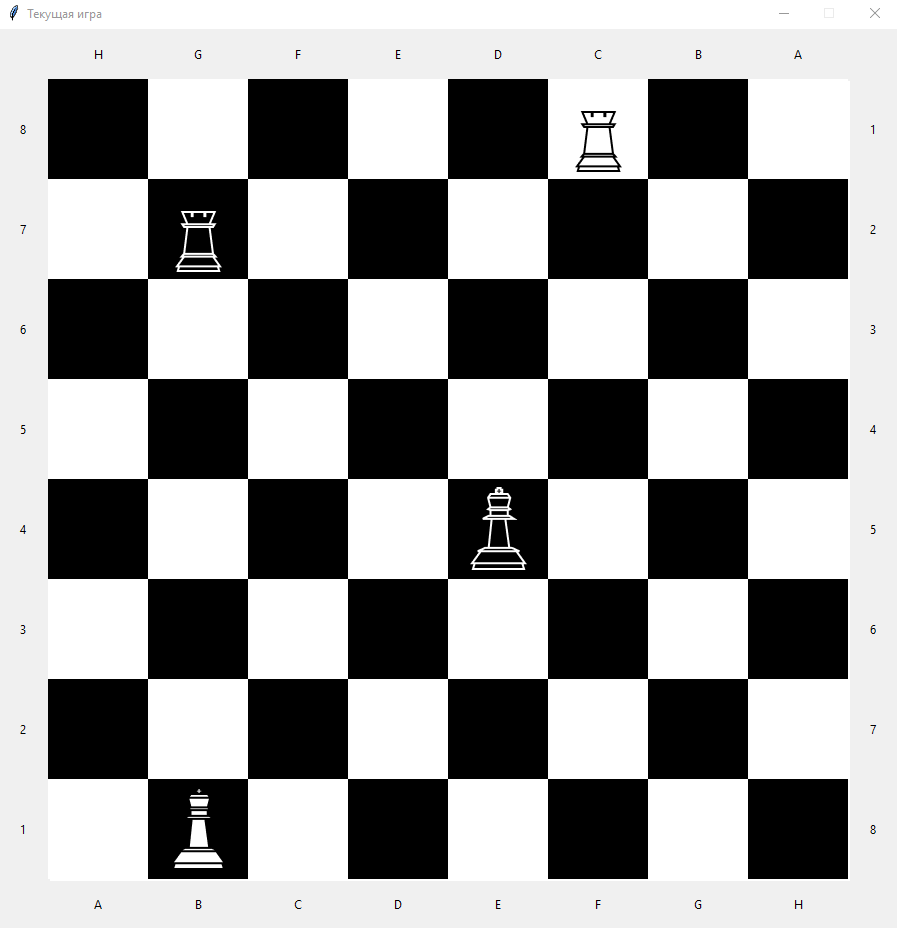


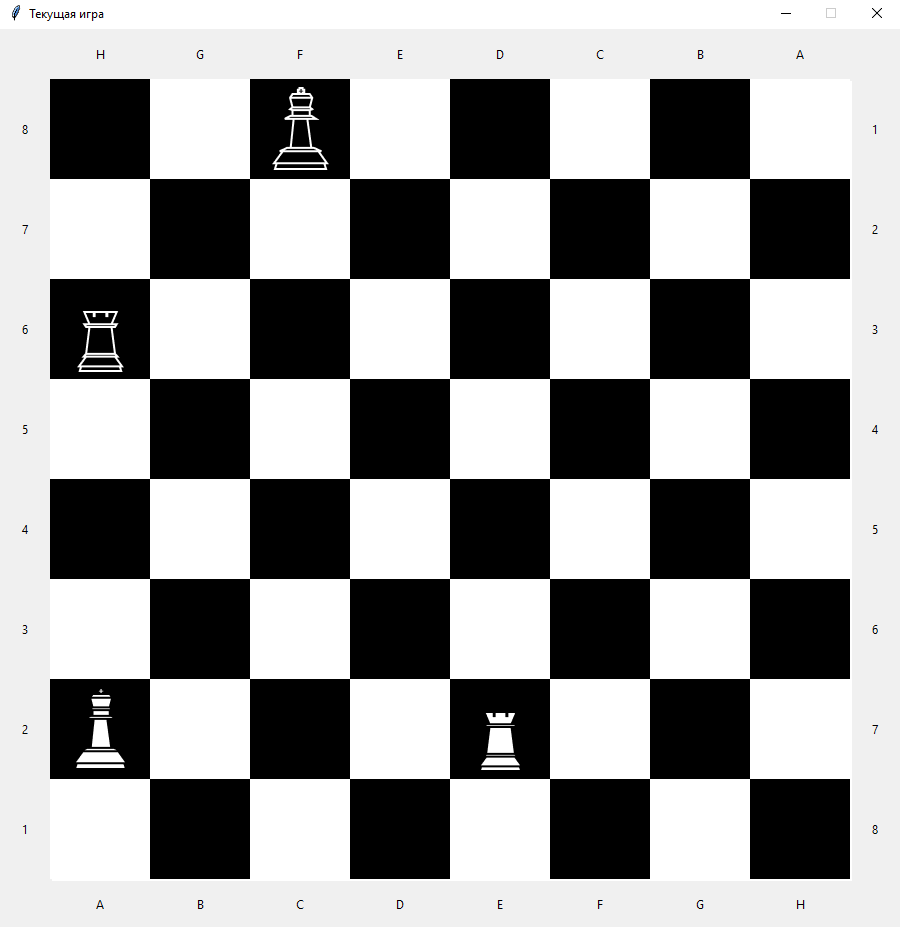


1. Отрисовка игрового поля

2.1. После нажатия на кнопку начала игры запускается отрисовка игрового поля и шахматных фигур. Шахматные фигуры расставляются случайным образом:

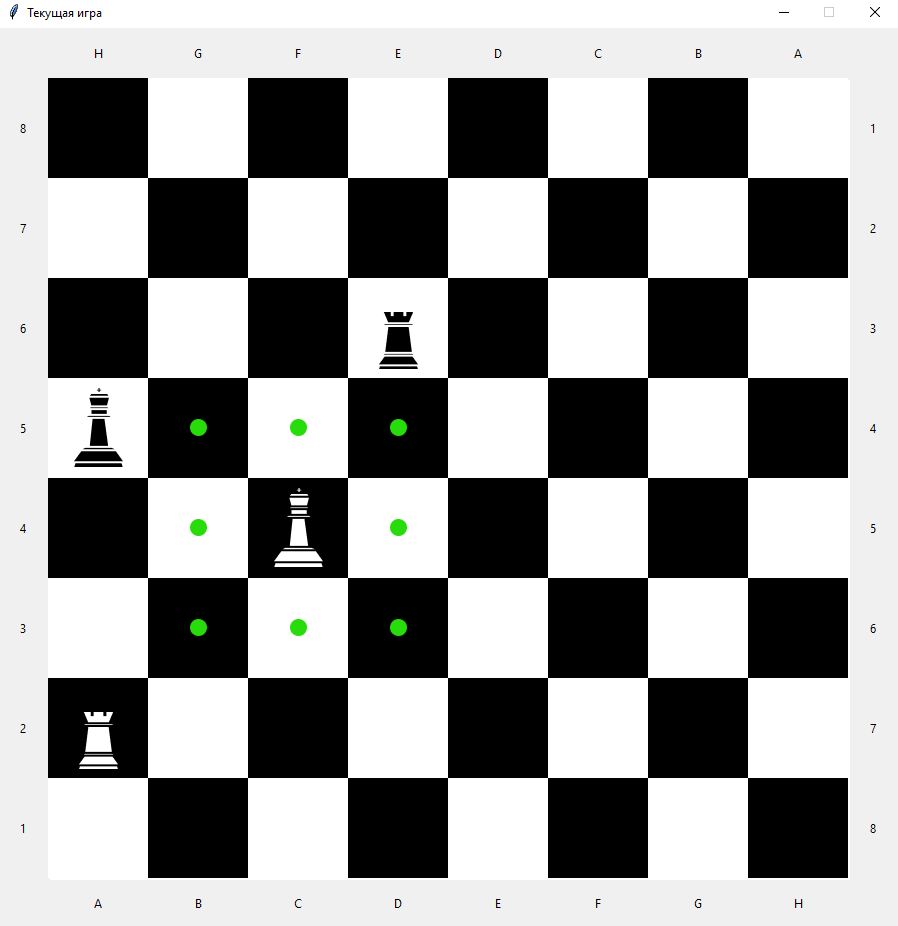


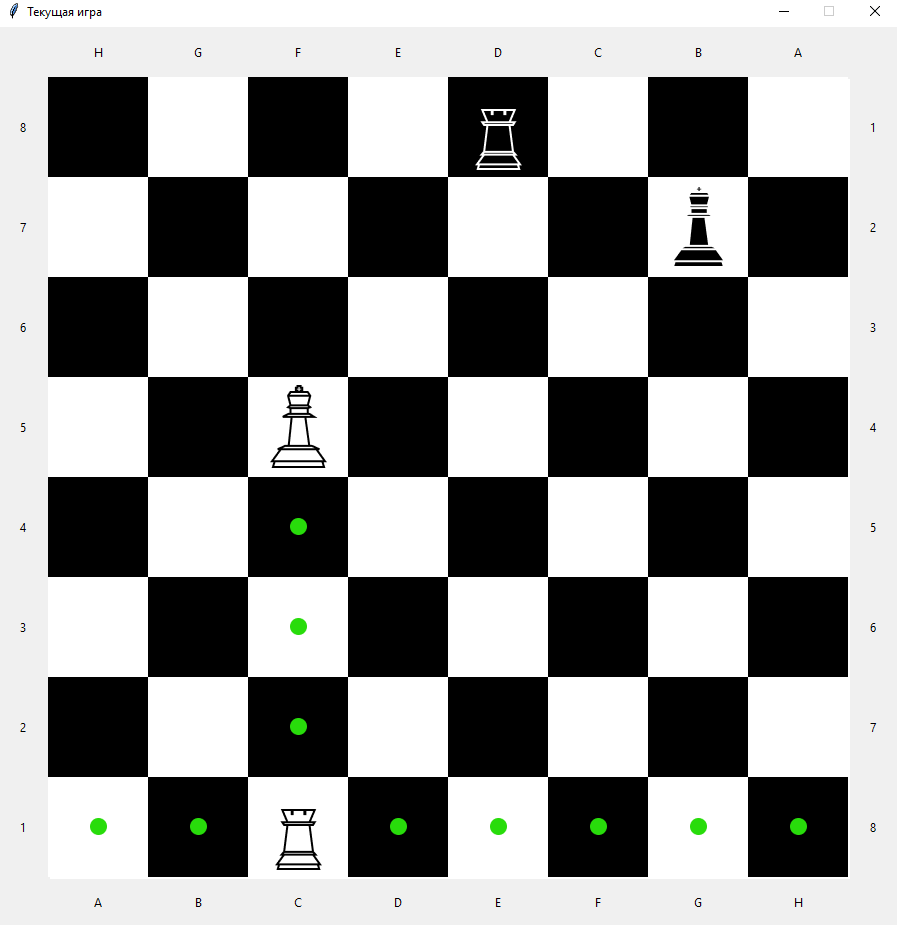




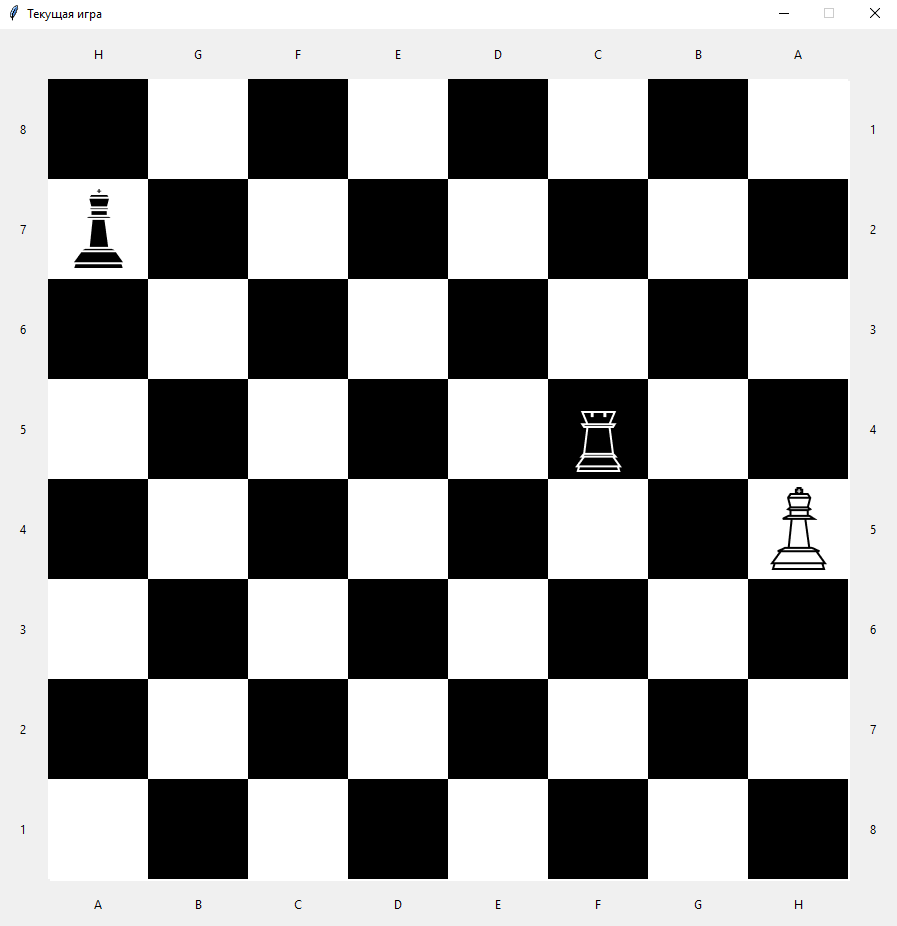
1. Ходы игроков

3.1. При нажатии на фигуру на поле появляются подсказки с возможными вариантами хода

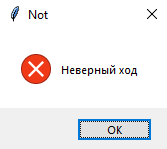




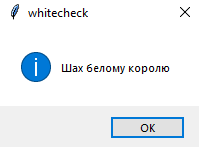
3.2. При “съедении” одной фигурой другой, фигура занимает место съеденной.



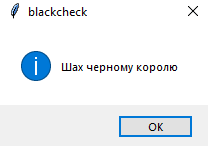
3.3. При нажатии на клетку, на которую нельзя сходить, появляется уведомление об ошибке:



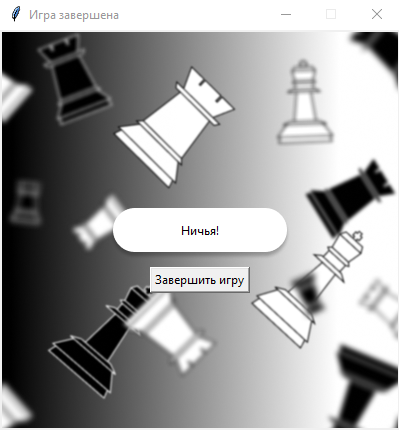
3.4. Если черная ладья ставит белому королю шах, появляется уведомление о шахе:



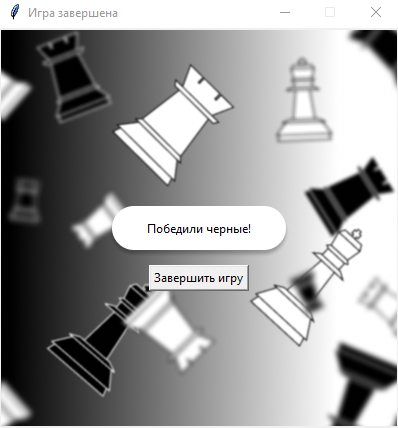
3.5. Если белая ладья ставит черному королю шах, появляется уведомление о шахе:



3.6. Если черная и белая ладьи оказываются съеденным, появляется уведомление о результате игры:



3.7. Если количество шахов для белого короля достигает 10, появляется уведомление о результате игры:



3.7. Если количество шахов для черного короля достигает 10, появляется уведомление о результате игры, аналогичное предыдущему пункту, но с победой белых.

Средства контроля ввода/вывода:

1. Проверка на пустое поле
2. Проверка на уже существующие логин и пароль
3. Проверка на длину логина и пароля
4. Проверка на авторизацию
5. Проверка на доступность хода
6. Проверка на шах
7. Проверка на результат игры

**2.2 Особенности реализации приложения**

Структуры данных, использованные в программе:

1. Массив моделирования игрового поля

chessplace = [[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0]]

1. Массив клеток шахматного поля

pole = [[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0],

[0,0,0,0,0,0,0,0]]

1. Массив для реализации Минимакса

weight = [[1,2,3,4,5,6,7,8],

[9,10,11,12,13,14,15,16],

[17,18,19,20,21,22,23,24],

[25,26,27,28,29,30,31,32],

[33,34,35,36,37,38,39,40],

[41,42,43,44,45,46,47,48],

[49,50,51,52,53,54,55,56],

[57,58,59,60,61,62,63,64]]

**3. Обращение к программе**

1. Tkinter – это кросс-платформенная событийно-ориентированная графическая библиотека Python, предназначенная для работы с библиотекой Tk. Библиотека Tk содержит компоненты графического интерфейса пользователя (GUI).
2. PIL - оригинальная библиотека Python для работы с изображениями
3. Модуль random предоставляет функции для генерации случайных чисел, букв, случайного выбора элементов последовательности.
4. Модуль tkinter.messagebox, предоставляющий доступ к готовым диалоговым окнам.

**4. Сообщения**

Сообщения, выдаваемые по результатам контроля корректности ввода/вывода:

1. Ошибка: "Должны быть введены данные" при проверке на пустое поле;
2. Ошибка: "Логин и пароль уже существуют" при проверке на уже существующие логин и пароль;
3. Ошибка: "Логин и пароль должны состоять не менее чем из 6 символов";
4. Информация: "Вы успешно зарегистрировались. Пожалуйста, авторизуйтесь" при успешной регистрации;
5. Ошибка: "Неверные логин или пароль" при проверке на авторизацию;
6. Ошибка: "Неверный ход" при выборе хода для белой фигуры;
7. Ошибка: "Сейчас ходят белые" при выборе фигуры для хода;
8. Информация: "Шах черному королю"
9. Информация: "Шах белому королю"

*Приложение 4*

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра «Измерительно-вычислительные комплексы»

**ТЕКСТ ПРОГРАММЫ**

на курсовую работу

по дисциплине «Алгоритмы и структуры данных»

ТемаКомпьютерная игра эндшпиль “Король, ладья - король, ладья” 

Р.02069337.22/2365-55 ТЗ-01

Листов 12

Исполнитель:

студент гр. ИСТбд-21

*Радченко Юлия Дмитриевна*

« » 2023

2023

from tkinter import \*

from tkinter import ttk

from PIL import Image, ImageTk

import random

import tkinter.messagebox as mb

# Создание окна регистрации и аторизации

root = Tk()

root.title("Регистрация и авторизация")

root.geometry("400x400")

frame = Frame(root)

au\_password = StringVar()

reg\_password = StringVar()

au\_login = StringVar()

reg\_login = StringVar()

bg = PhotoImage(file="backgrimage.png")

global poz\_hod\_x, poz\_hod\_y

poz\_hod\_x = 0; poz\_hod\_y = 0; hod = 1; ngo = 1; rook\_x = 0; rook\_y = 0; y\_last = 0; x\_last = 0; count\_wcheck = 0

count\_bcheck = 0; wcheck = 0; bcheck = 0; sum\_bel = 0; sum\_cher = 0

global chessplace # Массив моделирования игрового поля

chessplace = [[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0]]

global pole

pole = [[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0]]

global weight # Минимакс

weight = [[1, 2, 3, 4, 5, 6, 7, 8],

[9, 10, 11, 12, 13, 14, 15, 16],

[17, 18, 19, 20, 21, 22, 23, 24],

[25, 26, 27, 28, 29, 30, 31, 32],

[33, 34, 35, 36, 37, 38, 39, 40],

[41, 42, 43, 44, 45, 46, 47, 48],

[49, 50, 51, 52, 53, 54, 55, 56],

[57, 58, 59, 60, 61, 62, 63, 64]]

def entry\_regist\_and\_login(): # Ввод логина и пароля в окне регистрации и авторизации

# Регистрация

canvas = Canvas(root, width=400, height=400)

canvas.pack(fill="both", expand=True)

canvas.create\_image(0, 0, image=bg, anchor="nw")

canvas.create\_text(200, 25, text='Регистрация', font='30', fill="#FFFFFF")

canvas.create\_text(200, 60, text='Введите логин')

entry\_login = Entry(root, textvariable=reg\_login)

entry\_login = canvas.create\_window(140, 70, anchor="nw", window=entry\_login)

canvas.create\_text(200, 110, text='Введите пароль')

entry\_password = Entry(root, textvariable=reg\_password, show='\*')

entry\_password = canvas.create\_window(140, 120, anchor="nw", window=entry\_password)

button\_a = Button(root, text='Зарегистрироваться', command=reg\_login\_and\_password\_error, bg='#FFFFFF')

button\_a = canvas.create\_window(140, 160, anchor="nw", window=button\_a)

# Авторизация

canvas.create\_text(200, 220, text='Авторизация', font='30', fill="#FFFFFF")

canvas.create\_text(200, 250, text='Введите логин')

entry\_login = Entry(root, textvariable=au\_login)

entry\_login = canvas.create\_window(140, 260, anchor="nw", window=entry\_login)

canvas.create\_text(200, 300, text='Введите пароль')

entry\_password = Entry(root, textvariable=au\_password, show='\*')

entry\_password = canvas.create\_window(140, 310, anchor="nw", window=entry\_password)

button\_r = Button(root, text='Авторизироваться', command=au\_login\_and\_password\_error, bg='#FFFFFF')

button\_r = canvas.create\_window(145, 350, anchor="nw", window=button\_r)

def au\_login\_and\_password\_error(): # Проверка на ошибку авторизации

if (len(au\_login.get()) == 0 or len(au\_password.get()) == 0) or (

len(au\_login.get()) == 0 and len(au\_password.get()) == 0) or au\_login.get().count(' ') > 0 or \

au\_password.get().count(' ') > 0:

mb.showerror("Ошибка", "Должны быть введены данные")

else:

file = open('reg\_login.txt', 'r')

while TRUE:

line = file.readline()

line = line.split()

if len(line) == 0: # Ошибка авторизации

file.close()

mb.showerror("Ошибка", "Неверные логин или пароль")

break

elif line[0] == au\_login.get() and line[1] == au\_password.get(): # Успешная авторизация

file.close()

root.destroy()

window = Tk()

window.title("Успешная авторизация")

window.geometry("400x400")

canvas = Canvas(window, width=400, height=400)

canvas.pack(fill="both", expand=True)

img = PhotoImage(file="bckgrimg.png")

image = canvas.create\_image(200, 200, image=img)

canvas.create\_text(200, 200, text='Вы успешно авторизированы!', font='30')

button\_w = Button(window, text='Начать игру',

command=lambda: draw\_chessboard(window, white\_win, black\_win, gamedraw))

button\_w = canvas.create\_window(200, 250, window=button\_w)

window.grab\_set()

window.resizable(width=False, height=False)

window.mainloop()

break

def reg\_login\_and\_password\_error(): # Проверка на ошибку регистрации

# Проверка на пустое поле

if (len(reg\_login.get()) == 0 or len(reg\_password.get()) == 0) or (

len(reg\_login.get()) == 0 and len(reg\_password.get()) == 0) or reg\_login.get().count(' ') > 0 or \

reg\_password.get().count(' ') > 0:

mb.showerror("Ошибка", "Введены неверные данные")

# Проверка на длину логина и пароля

elif len(reg\_login.get()) <= 5 or len(reg\_password.get()) <= 5:

mb.showerror("Ошибка", "Логин и пароль должны состоять не менее чем из 6 символов")

else:

file = open('reg\_login.txt', 'r')

while TRUE:

line = file.readline()

line = line.split()

if len(line) > 0 and line[0] == reg\_login.get() and line[

1] == reg\_password.get(): # Проверка на совпадение логина и пароля

file.close()

mb.showerror("Ошибка", "Логин и пароль уже существуют")

break

else: # Успешня регистрация

file.close()

file = open('reg\_login.txt', 'a+')

file.write(reg\_login.get() + ' ' + reg\_password.get() + '\n')

file.close()

mb.showinfo("Успешная регистрация", "Вы успешно зарегистрировались. Пожалуйста, авторизуйтесь")

break

entry\_regist\_and\_login()

def draw\_chessboard(window, white\_win, black\_win, gamedraw): # Отрисовка игрового поля

global poz\_cher\_x, poz\_cher\_y, poz\_bel\_x, poz\_bel\_y

window.destroy()

global chessboard

chessboard = Tk()

chessboard.title("Текущая игра")

chessboard.geometry("900x900")

chessboard.resizable(width=False, height=False)

canvas = Canvas(chessboard)

canvas.pack(fill="both", expand=True)

# Буквенные обозначения по краям поля

# Снизу

canvas.create\_text(100, 875, text='A'), canvas.create\_text(200, 875, text='B'), canvas.create\_text(300, 875, text='C')

canvas.create\_text(400, 875, text='D'), canvas.create\_text(500, 875, text='E'), canvas.create\_text(600, 875, text='F')

canvas.create\_text(700, 875, text='G'), canvas.create\_text(800, 875, text='H')

# Сверху

canvas.create\_text(100, 25, text='H'), canvas.create\_text(200, 25, text='G'), canvas.create\_text(300, 25, text='F')

canvas.create\_text(400, 25, text='E'), canvas.create\_text(500, 25, text='D'), canvas.create\_text(600, 25, text='C')

canvas.create\_text(700, 25, text='B'), canvas.create\_text(800, 25, text='A')

# Численные обозначения по краям поля

# Справа

canvas.create\_text(875, 100, text='1'), canvas.create\_text(875, 200, text='2'), canvas.create\_text(875, 300, text='3')

canvas.create\_text(875, 400, text='4'), canvas.create\_text(875, 500, text='5'), canvas.create\_text(875, 600, text='6')

canvas.create\_text(875, 700, text='7'), canvas.create\_text(875, 800, text='8')

# Слева

canvas.create\_text(25, 100, text='8'), canvas.create\_text(25, 200, text='7'), canvas.create\_text(25, 300, text='6')

canvas.create\_text(25, 400, text='5'), canvas.create\_text(25, 500, text='4'), canvas.create\_text(25, 600, text='3')

canvas.create\_text(25, 700, text='2'), canvas.create\_text(25, 800, text='1')

# Создание шахматных клеток

canvas\_board = Canvas(chessboard, bg='white', width=800, height=800)

canvas\_board.place(x=50, y=50)

global color

worb = 0;

row = 0;

col = 0

for i in range(8): # Определение цвета клетки

for j in range(8):

worb += 1

if i % 2 == 0:

if worb % 2 == 0:

color = "white"

else:

color = "black"

elif i % 2 != 0:

if worb % 2 != 0:

color = "white"

else:

color = "black"

pole[i][j] = (Button(chessboard, bg=color, relief="flat",

command=lambda worb=worb, i=i, j=j: check\_figure(worb, i, j, 0)))

canvas\_board.create\_window(row, col, anchor=NW, window=pole[i][j], width=100, height=100)

row += 100

row = 0

col += 100

# Размещение шахматных фигур на поле

global img1, img2, img3, circle, img4, check\_shah\_bel, check\_shah\_cher, poz\_lad\_x\_bel, poz\_lad\_y\_bel, poz\_lad\_x\_cher, poz\_lad\_y\_cher

circle = PhotoImage(master=canvas\_board, file="greencircle.png") # Подсказка хода

img1 = PhotoImage(master=canvas\_board, file="img10.png") # Белая ладья

img2 = PhotoImage(master=canvas\_board, file="img20.png") # Черная ладья

img3 = PhotoImage(master=canvas\_board, file="img30.png") # Белый король

img4 = PhotoImage(master=canvas\_board, file="img40.png") # Черный король

# Начальное расположение

x\_wking = random.randint(0, 7)

y\_wking = random.randint(0, 7)

x\_bking = random.randint(0, 7)

y\_bking = random.randint(0, 7)

x\_wrook = random.randint(0, 7)

y\_wrook = random.randint(0, 7)

x\_brook = random.randint(0, 7)

y\_brook = random.randint(0, 7)

if chessplace[x\_wrook][y\_wrook] == 0: # Белая ладья

chessplace[x\_wrook][y\_wrook] = 1

pole[x\_wrook][y\_wrook]['image'] = img1

else:

chessplace[x\_wrook + 1][y\_wrook] = 1

pole[x\_wrook + 1][y\_wrook]['image'] = img1

if chessplace[x\_brook][y\_brook] == 0: # Черная ладья

chessplace[x\_brook][y\_brook] = -1

pole[x\_brook][y\_brook]['image'] = img2

else:

chessplace[x\_brook + 1][y\_brook] = -1

pole[x\_brook + 1][y\_brook]['image'] = img2

if chessplace[x\_wking][y\_wking] == 0: # Белый король

chessplace[x\_wking][y\_wking] = 2

pole[x\_wking][y\_wking]['image'] = img4

else:

chessplace[x\_wking + 1][y\_wking] = 2

pole[x\_wking + 1][y\_wking]['image'] = img4

if chessplace[x\_bking][y\_bking] == 0: # Черный король

chessplace[x\_bking][y\_bking] = -2

pole[x\_bking][y\_bking]['image'] = img3

else:

chessplace[x\_bking + 1][y\_bking] = -2

pole[x\_bking + 1][y\_bking]['image'] = img3

chessboard.mainloop()

def bot(): # Ходы бота

global poz\_kor\_x, poz\_kor\_y, poz\_lad\_bot\_x, poz\_lad\_bot\_y, poz\_kor\_bot\_x, poz\_kor\_bot\_y, poz\_lad\_x, poz\_lad\_y, count\_bcheck, wcheck, bcheck, sum\_bel

poz\_kor\_x = 0

poz\_kor\_y = 0

poz\_lad\_bot\_x = 0

poz\_lad\_bot\_y = 0

poz\_kor\_bot\_x = 0

poz\_kor\_bot\_y = 0

poz\_lad\_x = 0

poz\_lad\_y = 0

for i in range(0, 8): # Запоминание позиций фигур на поле

for j in range(0, 8):

if chessplace[i][j] == 2:

poz\_kor\_x = i

poz\_kor\_y = j

elif chessplace[i][j] == -1:

poz\_lad\_bot\_x = i

poz\_lad\_bot\_y = j

elif chessplace[i][j] == -2:

poz\_kor\_bot\_x = i

poz\_kor\_bot\_y = j

elif chessplace[i][j] == 1:

poz\_lad\_x = i

poz\_lad\_y = j

if sum\_cher == -1 and (poz\_lad\_x+1 == poz\_kor\_bot\_x or poz\_lad\_x-1 == poz\_kor\_bot\_x or poz\_lad\_y+1 == poz\_kor\_bot\_y or poz\_lad\_y-1 == poz\_kor\_bot\_y): # Если черная ладья съедена

check\_figure(0, poz\_kor\_bot\_x, poz\_kor\_bot\_y, 1)

check\_figure(0, poz\_lad\_x, poz\_lad\_y, 1)

sum\_bel = -1

bcheck = 0

elif sum\_cher == -1:

check\_figure(0, poz\_kor\_bot\_x, poz\_kor\_bot\_y, 1)

check\_figure(0, poz\_hod\_x, poz\_hod\_y, 1)

elif (poz\_lad\_bot\_x == poz\_lad\_x or poz\_lad\_bot\_y == poz\_lad\_y) and (

poz\_lad\_bot\_x != poz\_kor\_bot\_x or poz\_lad\_bot\_y != poz\_kor\_bot\_y): # Если есть шах для черного короля и две ладьи на одной прямой

check\_figure(0, poz\_lad\_bot\_x, poz\_lad\_bot\_y, 1)

check\_figure(0, poz\_lad\_x, poz\_lad\_y, 1)

sum\_bel = -1

elif bcheck == 1 and (poz\_lad\_x+1 == poz\_kor\_bot\_x or poz\_lad\_x-1 == poz\_kor\_bot\_x or poz\_lad\_y+1 == poz\_kor\_bot\_y or poz\_lad\_y-1 == poz\_kor\_bot\_y): # Если есть шах для черного короля и белая ладья на соседней клетке

check\_figure(0, poz\_kor\_bot\_x, poz\_kor\_bot\_y, 1)

check\_figure(0, poz\_lad\_x, poz\_lad\_y, 1)

sum\_bel = -1

bcheck = 0

elif bcheck == 1: # Если есть шах для черного короля

check\_figure(0, poz\_kor\_bot\_x, poz\_kor\_bot\_y, 1)

check\_figure(0, poz\_hod\_x, poz\_hod\_y, 1)

bcheck = 0

elif bcheck == 0: # Если нет шаха для черного короля

check\_figure(0, poz\_lad\_bot\_x, poz\_lad\_bot\_y, 1)

check\_figure(0, poz\_hod\_x, poz\_hod\_y, 1)

result(white\_win, black\_win, gamedraw)

return

def check\_hod(x, y, k, clear, bot): # Проверка на правильность хода

global poz\_hod\_x, poz\_hod\_y

max\_weight = 0

if clear == 0: # Установка подсказок

if k == 1 or k == -1: # Ладья

for i in range(y + 1, 8): # Ход вправо

if (i >= 0 and i < 8) and pole[x][i]['image'] == '':

if bot == 1:

if i == poz\_kor\_y:

poz\_hod\_x = x

poz\_hod\_y = i

return

else:

pole[x][i]['image'] = circle

elif (i >= 0 and i < 8) and pole[x][i]['image'] != '':

break

for i1 in range(y - 1, 0 - 1, -1): # Ход влево

if (i1 >= 0 and i1 < 8) and pole[x][i1]['image'] == '':

if bot == 1:

if i1 == poz\_kor\_y:

poz\_hod\_x = x

poz\_hod\_y = i1

return

else:

pole[x][i1]['image'] = circle

elif (i1 >= 0 and i1 < 8) and pole[x][i1]['image'] != '':

break

for i2 in range(x + 1, 8): # Ход вверх

if (i2 >= 0 and i2 < 8) and pole[i2][y]['image'] == '':

if bot == 1:

if i2 == poz\_kor\_x:

poz\_hod\_x = i2

poz\_hod\_y = y

return

else:

pole[i2][y]['image'] = circle

elif (i2 >= 0 and i2 < 8) and pole[i2][y]['image'] != '':

break

for i3 in range(x - 1, 0 - 1, -1): # Ход вниз

if (i3 >= 0 and i3 < 8) and pole[i3][y]['image'] == '':

if bot == 1:

if i3 == poz\_kor\_x:

poz\_hod\_x = i3

poz\_hod\_y = y

return

else:

pole[i3][y]['image'] = circle

elif (i3 >= 0 and i3 < 8) and pole[i3][y]['image'] != '':

break

elif k == 2 or k == -2: # Король

for i in range(x - 1, x + 2):

for j in range(y - 1, y + 2):

if ((i >= 0 and i < 8) and (j >= 0 and j < 8)) and pole[i][j]['image'] == '':

if bot == 1:

if (i != poz\_lad\_x and j != poz\_lad\_y) and max\_weight < weight[i][j]:

max\_weight = weight[i][j]

poz\_hod\_x = i

poz\_hod\_y = j

else:

pole[i][j]['image'] = circle

elif clear == 1: # Удаление подсказок

if k == 1: # Ладья

for i in range(y + 1, 8): # Ход вправо

if (i >= 0 and i < 8) and chessplace[x][i] == 0:

pole[x][i]['image'] = ''

for i1 in range(y - 1, 0 - 1, -1): # Ход влево

if (i1 >= 0 and i1 < 8) and chessplace[x][i1] == 0:

pole[x][i1]['image'] = ''

for i2 in range(x + 1, 8): # Ход вверх

if (i2 >= 0 and i2 < 8) and chessplace[i2][y] == 0:

pole[i2][y]['image'] = ''

for i3 in range(x - 1, 0 - 1, -1): # Ход вниз

if (i3 >= 0 and i3 < 8) and chessplace[i3][y] == 0:

pole[i3][y]['image'] = ''

elif k == 2: # Король

for i in range(x - 2, x + 2):

for j in range(y - 2, y + 2):

if ((i >= 0 and i < 8) and (j >= 0 and j < 8)) and chessplace[i][j] == 0:

pole[i][j]['image'] = ''

def check\_figure(i, x, y, boti): # Ходы

global hod, ngo, x\_last, y\_last, last\_image, rook\_x, rook\_y, a, b, sum\_bel, sum\_cher, blackcount, poz\_lad\_bot\_x, poz\_lad\_bot\_y

if ngo == 1 and pole[x][y]['image'] != '': # Первое нажатие

if (chessplace[x][y] == -1 or chessplace[x][y] == -2) and boti == 0:

mb.showerror(title="error", message="Сейчас ходят белые")

else:

if chessplace[x][y] == 1:

rook\_x = x

rook\_y = y

ngo = 2

hod += 1

last\_image = pole[x][y]['image']

x\_last = x

y\_last = y

check\_hod(x, y, chessplace[x][y], 0, boti)

if boti == 1:

ngo = 2

return

elif x == x\_last and y == y\_last:

check\_hod(x, y, chessplace[x][y], 1, boti)

ngo = 1

return

elif ngo == 2 and pole[x][y]['image'] != '' and chessplace[x][y] != -2 and chessplace[x][y] != 2 and boti == 0: # Второе нажатие

if chessplace[x][y] == -1:

sum\_cher = -1

chessplace[x][y] = chessplace[x\_last][y\_last]

chessplace[x\_last][y\_last] = 0

pole[x][y]['image'] = last\_image

pole[x\_last][y\_last]['image'] = ''

check\_hod(x\_last, y\_last, chessplace[x][y], 1, boti)

check\_and\_mate()

ngo = 1

bot()

return

elif ngo == 2 and (pole[x][y]['image'] == '' or pole[x][y]['image'] != '') and boti == 1:

chessplace[x][y] = chessplace[x\_last][y\_last]

chessplace[x\_last][y\_last] = 0

pole[x][y]['image'] = last\_image

pole[x\_last][y\_last]['image'] = ''

check\_and\_mate()

ngo = 1

return

elif ngo == 2 and pole[x][y]['image'] == '' and boti == 0:

mb.showerror(title="error", message="Неправильный ход")

def result(white\_win, black\_win, gamedraw): # Определение результата игры

global end

if count\_wcheck == 10: # 10 шахов, победа черных

black\_win(chessboard, end\_of\_play, result)

elif count\_bcheck == 10: # 10 шахов, победа белых

white\_win(chessboard, end\_of\_play, result)

elif sum\_cher == -1 and sum\_bel == -1: # Съедены белая и черная ладьи, ничья

gamedraw(chessboard, end\_of\_play, result)

def white\_win(chessboard, end\_of\_play, result): # Вывод результата игры, победа белых

chessboard.destroy()

end = Tk()

end.title("Игра завершена")

end.geometry("400x400")

canvas = Canvas(end, width=400, height=400)

canvas.pack(fill="both", expand=True)

img = PhotoImage(file="end.png")

image = canvas.create\_image(200, 200, image=img)

canvas.create\_text(200, 200, text='Победили белые!')

button\_e = Button(end, text='Завершить игру', command=lambda: end\_of\_play(end))

button\_e = canvas.create\_window(200, 250, window=button\_e)

end.grab\_set()

end.resizable(width=False, height=False)

end.mainloop()

def black\_win(chessboard, end\_of\_play, result): # Вывод результата игры, победа черных

chessboard.destroy()

end = Tk()

end.title("Игра завершена")

end.geometry("400x400")

canvas = Canvas(end, width=400, height=400)

canvas.pack(fill="both", expand=True)

img = PhotoImage(file="end.png")

image = canvas.create\_image(200, 200, image=img)

canvas.create\_text(200, 200, text='Победили черные!')

button\_e = Button(end, text='Завершить игру', command=lambda: end\_of\_play(end))

button\_e = canvas.create\_window(200, 250, window=button\_e)

end.grab\_set()

end.resizable(width=False, height=False)

end.mainloop()

def gamedraw(chessboard, end\_of\_play, result): # Вывод результата игры, ничья

chessboard.destroy()

end = Tk()

end.title("Игра завершена")

end.geometry("400x400")

canvas = Canvas(end, width=400, height=400)

canvas.pack(fill="both", expand=True)

img = PhotoImage(file="end.png")

image = canvas.create\_image(200, 200, image=img)

canvas.create\_text(200, 200, text='Ничья!')

button\_e = Button(end, text='Завершить игру', command=lambda: end\_of\_play(end))

button\_e = canvas.create\_window(200, 250, window=button\_e)

end.grab\_set()

end.resizable(width=False, height=False)

end.mainloop()

def end\_of\_play(end): # Завершение игры

end.destroy()

def check\_and\_mate(): # Проверка на шах

global count\_wcheck, count\_bcheck, bcheck, wcheck

for x in range(0, 8):

for j in range(0, 8):

if chessplace[x][j] == 2: # Для белого короля

# По горизонтали

whitefigure = 0

for i in range(j + 1, 8):

if (i >= 0 and i < 8):

if chessplace[x][i] == 1 or chessplace[x][i] == -2:

whitefigure += 1

elif chessplace[x][i] == -1 and whitefigure == 0:

mb.showinfo(title="whitecheck", message="Шах белому королю")

count\_wcheck += 1

wcheck = 1

result(white\_win, black\_win, gamedraw)

whitefigure = 0

for i1 in range(j - 1, 0 - 1, -1):

if (i1 >= 0 and i1 < 8):

if chessplace[x][i1] == 1 or chessplace[x][i1] == -2:

whitefigure += 1

elif chessplace[x][i1] == -1 and whitefigure == 0:

mb.showinfo(title="whitecheck", message="Шах белому королю")

count\_wcheck += 1

wcheck = 1

result(white\_win, black\_win, gamedraw)

# По вертикали

whitefigure = 0

for i2 in range(x + 1, 8):

if (i2 >= 0 and i2 < 8):

if chessplace[i2][j] == 1 or chessplace[i2][j] == -2:

whitefigure += 1

elif chessplace[i2][j] == -1 and whitefigure == 0:

mb.showinfo(title="whitecheck", message="Шах белому королю")

count\_wcheck += 1

wcheck = 1

result(white\_win, black\_win, gamedraw)

whitefigure = 0

for i3 in range(x - 1, 0 - 1, -1):

if (i3 >= 0 and i3 < 8):

if chessplace[i3][j] == 1 or chessplace[i3][j] == -2:

whitefigure += 1

elif chessplace[i3][j] == -1 and whitefigure == 0:

mb.showinfo(title="whitecheck", message="Шах белому королю")

count\_wcheck += 1

wcheck = 1

result(white\_win, black\_win, gamedraw)

elif chessplace[x][j] == -2: # Для черного короля

# По горизонтали

blackfigure = 0

for i in range(j + 1, 8):

if (i >= 0 and i < 8):

if chessplace[x][i] == -1 or chessplace[x][i] == 2:

blackfigure += 1

elif chessplace[x][i] == 1 and blackfigure == 0:

mb.showinfo(title="blackcheck", message="Шах черному королю")

count\_bcheck += 1

bcheck = 1

result(white\_win, black\_win, gamedraw)

blackfigure = 0

for i1 in range(j - 1, 0 - 1, -1):

if (i1 >= 0 and i1 < 8):

if chessplace[x][i1] == -1 or chessplace[x][i1] == 2:

blackfigure += 1

elif chessplace[x][i1] == 1 and blackfigure == 0:

mb.showinfo(title="blackcheck", message="Шах черному королю")

count\_bcheck += 1

bcheck = 1

result(white\_win, black\_win, gamedraw)

# По вертикали

blackfigure = 0

for i2 in range(x + 1, 8):

if (i2 >= 0 and i2 < 8):

if chessplace[i2][j] == -1 or chessplace[i2][j] == 2:

blackfigure += 1

elif chessplace[i2][j] == 1 and blackfigure == 0:

mb.showinfo(title="blackcheck", message="Шах черному королю")

count\_bcheck += 1

bcheck = 1

result(white\_win, black\_win, gamedraw)

blackfigure = 0

for i3 in range(x - 1, 0 - 1, -1):

if (i3 >= 0 and i3 < 8):

if chessplace[i3][j] == -1 or chessplace[i3][j] == 2:

blackfigure += 1

elif chessplace[i3][j] == 1 and blackfigure == 0:

mb.showinfo(title="blackcheck", message="Шах черному королю")

count\_bcheck += 1

bcheck = 1

result(white\_win, black\_win, gamedraw)

root.resizable(width=False, height=False)

root.mainloop()