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 логическая игра Реверси

**Инв. № подл.**

**Подп. и дата**

**Взам. инв. №**

**Инв. № дубл.**

**Подп. и дата**

Р.02069337.22/2411-12 ТЗ-01

Листов 30

Руководитель разработки:

кандидат технических наук, доцент кафедры «Измерительно-вычислительные комплексы»,  
Шишкин Вадим Викторинович

« » 2023 г.

Исполнитель:

студент гр. ИСТбд-21

*Коцан Иван Владимирович*

« » 2023 г.

2023

Содержание

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

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

Пояснительная записка...…………………………………………….....7

Руководство пользователя……………………………….…………....12

Текст программы…..………………………………………………….....19

**Аннотация**

*Задание на курсовую работу:* разработка компьютерной логической игры «Реверси».

*Суть задания:* необходимо разработать приложение, которое соответствует правилам настольной игры «Реверси». Оно должно представлять собой окно, представляющую шахматную доску с интуитивно понятные интерфейсом и основным функционалом.

*Ключевые функции приложения:*

1. Авторизация/регистрация пользователя.
2. Отображение игрового поля.
3. Проверка ходов игрока.
4. Ходы искусственного интеллекта.
5. Объявление результатов игры.

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 логическая игра Реверси

**Инв. № подл.**

**Подп. и дата**

**Взам. инв. №**

**Инв. № дубл.**

**Подп. и дата**

Р.02069337.22/2411-12 ТЗ-01

Листов 3

Исполнитель:

студент гр. ИСТбд-21

*Коцан Иван Владимирович*

« » 2023 г.

2023

**Введение**

Название разрабатываемого приложения — "Reversi". Это приложение представляет собой электронную версию игры "Реверси", доступную на персональных компьютерах.

Правила: в процессе хода игрок должен расположить свою шашку так, чтобы одна или несколько шашек противника были окружены: по диагонали, вертикали или горизонтали. Если у игрока нет возможности сделать ход так, чтобы окружить одну или несколько шашек противника, ход переходит к другому игроку.

Приложение должно содержать окно авторизации/регистрации пользователя, главное окно, на котором находится игровое поле и информационное поле, где указывается текущее количество очков у игроков, а также подсказки по игре.

**1. Основания для разработки**

В качестве оснований для разработки указывается учебный план направления 09.03.02 «Информационные системы и технологии» и распоряжение по факультету.

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

**2.1. Функциональное назначение**

Требуется разработать однопользовательское десктопное приложение по игре реверси 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

Приложение должно соответствовать следующим правилам игры.

Игра ведется между двумя соперниками (пользователь-компьютер или пользователь-пользователь) шашками на квадратном поле размером 8х8 клеток. У шашки с одной стороны белый цвет, с другой чёрный. В начальной позиции у каждого игрока по 2 шашки расположены в середине игрового поля по диагонали. На руках по 30 шашек.

Игроки совершают ходы поочередно. В процессе хода игрок должен расположить свою шашку так, чтобы одна или несколько шашек противника были окружены: по диагонали, вертикали или горизонтали. Если у игрока нет возможности сделать ход так, чтобы окружить одну или несколько шашек противника, ход переходит к другому игроку.

Цель игры – как можно больше окружить фишек противника. Игра считается завершённой, если ни один из игроков не может сделать ход. Игрок, у которого на поле больше фишек своего цвета, чем у противника – считается победителем, у которого меньше – проигравшим. Если количество фишек на поле одинаковое – то объявляется ничья.

2.2.3 Требования к организации информационного обеспечения, входных и выходных данных

**2.3 Требования к надёжности**

* Приложение должно быть устойчивым к возможным сбоям, ошибкам и исключительным ситуациям.
* Обработка исключительных ситуаций и ошибок с помощью механизмов обработки исключений.
* Предотвращение возможных утечек памяти и ресурсов.

**2.4 Требования к информационной и программной совместимости**

* OC Windows
* Версия Python: 3.11
* Среда разработки: PyCharm Community Edition 2022.3.2

**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

Требования к условиям хранения не предъявляются

2.6 3 Сроки хранения

Срок хранения – до июля 2024 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.

**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 логическая игра Реверси

**Пояснительная записка**

Р.02069337.22/2411-12 ПЗ-01

**Инв. № подл.**

**Подп. и дата**

**Взам. инв. №**

**Инв. № дубл.**

**Подп. и дата**

Листов 5

**Исполнитель**:

студент гр. ИСТбд-21

*Коцан Иван Владимирович*

« » 2023 г.

2023

**Введение**

Название разрабатываемого приложения — "Reversi". Это приложение представляет собой электронную версию игры "Реверси", доступную на персональных компьютерах.

При разработке "Reversi" был выбран подход, основанный на математической модели игры "Реверси". Математическая модель позволяет точно определить логику ходов, правила игры и алгоритмы принятия решений компьютерного противника. Такой подход обеспечивает точность и надежность реализации игры, а также позволяет создать интеллектуального компьютерного противника, способного представить вызов даже опытным игрокам.

Приложение обладает удобным и интуитивно понятным интерфейсом, позволяющим игрокам легко размещать свои фишки на игровом поле и применять стратегические решения.

**1. Проектная часть**

**1.1 Постановка задачи на разработку приложения**

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

**1.2 Архитектура и алгоритмы**

1.2.1. Архитектура

Классы и структура данных:

Authorization: этот класс представляет окно авторизации/регистрации пользователя. Он содержит методы для отображения окна, проверки данных о зарегистрированных пользователях и последующей авторизации, а также метод регистрации пользователя.

ReversiGame: этот класс представляет игру «Реверси». Он содержит методы для управления игровым процессом, хранит состояние игрового поля.

**1.3 Тестирование**



1.3.1 Описание отчета о тестировании

Отчет о тестировании игры "Реверси" предоставляет обзор текущего состояния игры и качества ее реализации. Он включает информацию о проведенных функциональных тестах, тестировании пользовательского интерфейса, проверке граничных значений, совместимости, производительности и безопасности. Каждый из этих аспектов тестирования подробно описывается в отчете.

Основная цель отчета о тестировании игры "Реверси" - предоставить полную картину о ее качестве и выявленных проблемах. Отчет содержит рекомендации по исправлению проблем и улучшению функциональности игры.

1.3.2 Цель тестирования

Проверить функциональность, корректность ввода/вывода и общую стабильность приложения "Реверси".

1.3.3 Методика тестирования

Модульное тестирование: в этом методе можно проводится тестирование отдельных модулей игры, таких как логика ходов, правила игры, алгоритмы принятия решений компьютерного противника и другие важные компоненты. Это позволит убедиться в правильной работе каждого модуля и их соответствии требованиям.

Интеграционное тестирование: здесь проверяется взаимодействие различных модулей игры между собой. Например, как правильно обновляется игровое поле при ходе игрока или компьютера, как корректно обновляется счет игры и т.д. Целью является выявление возможных ошибок во взаимодействии компонентов игры.

Функциональное тестирование: этот метод позволяет проверить функциональность игры в соответствии с требованиями. Проверить правильность подсчета очков, корректность применения правил игры и другие функциональные аспекты.

Графическое тестирование: этот метод предполагает проверку корректности отображения графического интерфейса игры. Это включает проверку расположения элементов на экране, соответствие цветов и шрифтов, и других визуальных аспектов игры.

1.3.4 Проведенные тесты

Тестирование функциональности:

- Цель: проверить основную функциональность игры "Реверси".

- Шаги выполнения:

1. Открыть приложение "Реверси".

2. Зарегистрировать нового пользователя.

3. Начать игру.

4. Выполнить несколько ходов.

5. Завершить игру.

Ожидаемые результаты:

- Пользователь успешно зарегистрирован.

- Игра успешно запускается.

- Игровое поле отображается корректно.

- Игрок может совершать ходы в соответствии с правилами игры.

- Результаты игры отображаются правильно.

- Фактические результаты: Все ожидаемые результаты подтверждены.

Тестирование корректности ввода/вывода:

Цель: проверить корректность обработки пользовательского ввода и вывода информации.

- Шаги выполнения:

1. Оставить окно регистрации пустым.

2. Выбрать клетку во время хода, не соответствующую правилам игры.

Ожидаемые результаты:

- Если оставить поля в окне регистрации пустыми, то выводится сообщение об ошибке.

- При выборе клетки во время хода, не соответствующей правилам игры, ход не производится.

Фактические результаты: все ожидаемые результаты подтверждены.

1.3.5 Выводы

Приложение "Реверси" демонстрирует стабильную работу и правильную функциональность в большинстве случаев.

Необходимо добавить подсказку, когда игрок пытается сделать ход, не соответствующий правилам игры.

**2. Источники, использованные при разработке**

1. OTHELLO: FROM BEGINNER TO MASTER [Электронный ресурс] URL: <http://othello.federation.free.fr/livres/beginner-Randy-Fang.pdf>

2. Реверси – стратегия игры [Электронный ресурс] URL: <https://reversi.ucoz.ru/>

**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 логическая игра Реверси

**Руководство программиста**

Р.02069337.22/2411-12 РП-01

**Инв. № подл.**

**Подп. и дата**

**Взам. инв. №**

**Инв. № дубл.**

**Подп. и дата**

Листов 19

Исполнитель:

студент гр. ИСТбд-21

*Коцан Иван Владимирович*

« » 2023 г.

2023

**1. Назначение и условия применения программы**

**1.1 Назначение и функции, выполняемые приложением**

Функциональное назначение: разработка приложения для игры "Реверси" с целью предоставления пользователю возможности играть в эту стратегическую настольную игру на своем устройстве.

Свод правил игры "Реверси":

* Игра проводится на игровом поле размером 8x8 клеток.
* У каждого игрока есть фишки одного цвета, обычно черного и белого.
* Игроки ходят по очереди, ставя фишки на свободные клетки поля.
* Ход делается путем размещения фишки на клетке таким образом, чтобы между новой фишкой и уже установленными фишками противника находились фишки этого же цвета.
* Фишки противника, попавшие между новой фишкой и другими фишками текущего игрока, меняют цвет на цвет текущего игрока.
* Цель игры - иметь на игровом поле больше фишек своего цвета, чем у противника, когда все клетки заполнены.

Общая характеристика функциональных возможностей приложения:

1. Игра с компьютерным противником: приложение предоставляет возможность играть против компьютерного противника. Компьютерный противник может использовать различные алгоритмы и стратегии для принятия решений и выполнения ходов.
2. Игра с другими игроками: Приложение позволяет играть в многопользовательском режиме, где пользователи могут играть против друг друга на одном устройстве или по сети.

**1.2 Условия, необходимые для использования приложения**

Требования к операционной системе: Windows

Требования к платформе: настольный компьютер

**2. Характеристики программы**

**2.1 Характеристики приложения**

Количество значимых строк программного кода: 271.

Библиотеки, используемые при написании кода:

os – для записи данных о зарегистрированных пользователях;

tkinter – для отображения окон, игрового поля и других графических интерфейсов в программе.

После запуска приложения у пользователя появляется окно авторизации/регистрации (рис.1).

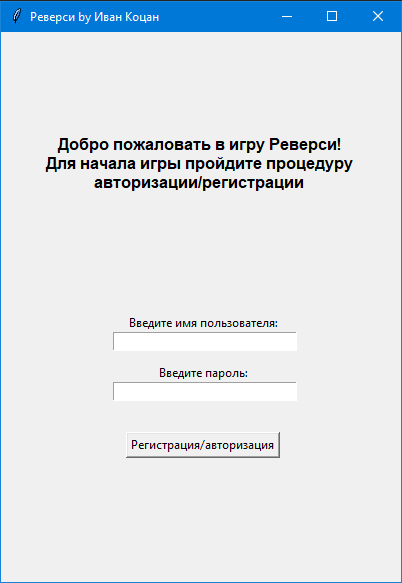


Рис.

В появившемся окне необходим ввести своё имя пользователя. Если пользователь не зарегистрирован, ему необходимо придумать своё имя.  
Во втором поле необходимо ввести пароль, указанный при регистрации. Если аккаунт пользователя ещё не создан, то необходимо придумать пароль.

После успешной авторизации/регистрации появляется окно (рис.2), где предлагается начать игру.

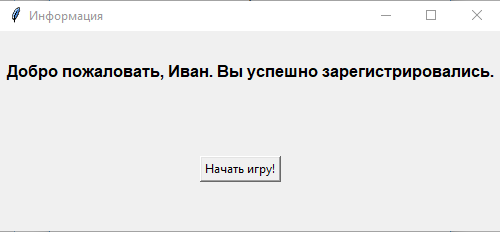


Рис.

После нажатия кнопки «Начать игру!» появляется окно (рис. 3): на котором расположено игровое поле размеров 8x8 клеток, а также блок с информацией об игре, где указан текущее количество очков у каждого из игроков и кто сейчас делает ход. В самом низу окна располагается кнопка «Новая игра», после нажатия которой текущая игра завершается и шашки возвращаются на исходное место.

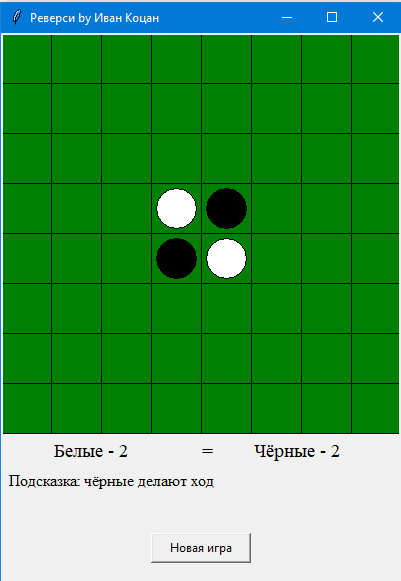


Рис.

Чтобы сделать ход, необходимо нажать на соответствующую клетку игрового поля. После хода пользователя, автоматически делает ход искусственный интеллект.

Когда количество очков у игроков отличается, то цвет текста с названием цвета шашек потенциального победителя становится зелёным, а также меняется знак неравенства (рис. 4).



Рис.

Если у игрока, который должен сделать ход, нет свободных ходов, то всплывает соответствующая подсказка, в которой говорится, что другой игрок делает ход повторно.

Если ни один из игроков не может сделать ход, но не все клетки игрового поля заняты шашками, то игра оканчивается и выбирается победитель, у которого больше очков. Вместе с этим всплывает соответствующая подсказка о том, что игра окончена досрочно.

Игра считается завершённой, когда ни один из игроков не может сделать ход и все клетки игрового поля заняты шашками. В этом случае также происходит подсчёт очков и выбирается победитель. В нижней части окна появляется подсказка об окончании игры и указывается победитель (рис. 5).

Если при окончании количество очков у игроков совпадает, то считается, что игра окончена в ничью.

После окончания игры пользователь может начать игру заново, нажав кнопку «Новая игра» в нижней части окна, или закрыть окно для выхода из программы.

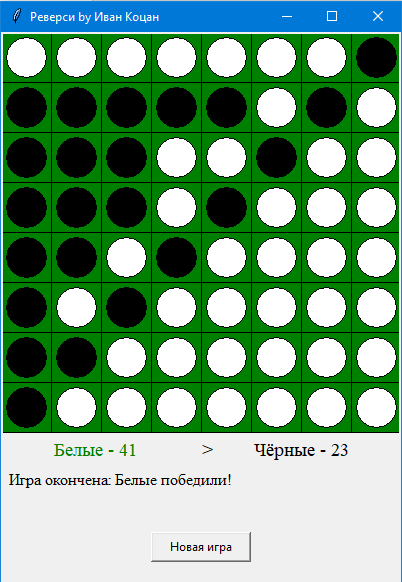


Рис.

**2.2 Особенности реализации приложения**

Структура данных.

Игровое поле: для представления игрового поля размером 8x8 клеток, используется двумерный массив или матрицу. Каждая клетка представлена значением, указывающим на цвет фишки (0 – пустая клетка, 1 – чёрная шашка, 2 – белая шашка). Использование массива обеспечивает эффективный доступ к элементам и удобство при выполнении операций на игровом поле.

Альтернативный вариант: вместо двумерного массива для игрового поля можно использовать одномерный массив с преобразованием индекса клетки в двумерные координаты. Это может упростить некоторые операции, но может быть менее интуитивно понятным при работе с матрицей.

**3. Обращение к программе**

1. Класс «Авторизация»:

**-** методы:

-\_\_init\_\_(): создание окна авторизации/регистрации пользователя.

-регист(): произвести запись данных зарегистрированного пользователя.

-нажать(): убрать информационное окно, в случае успешной авторизации отобразить окно с игровым полем.

-показать\_информацию(текст, кнопка): отобразить информационное окно с параметрами текст (текст, содержащий информацию) и кнопка (название кнопки в нижнем части окна информации).

2. Класс «РеверсиИгра»:

- методы:

-\_\_init\_\_(): создание окна с игровым полем и информацией в нижней части окна.

-новая\_игра(): вернуть все фишки в исходное место, сбросить очки у игроков.

-нарисовать\_доску(): рисование доски и фишек, расположенных на ней.

-нажатие(): обработка действий при нажатии на клетку игрового поля.

-проверка\_правильности\_хода: проверяет ход игрока, при нажатии на клетку.

-проверка\_хода: проверяет, пустая клетка или нет.

-сделать\_ход: делает ход, меняет фишки противника на свои, в случае их окружения.

Библиотека os:

Библиотека используется для записи данных при регистрации пользователя, а также проверки данных при авторизации пользователя. Функции, используемые в коде: os.path.exists (проверка, существует ли файл).

Библиотека tkinter:

Библиотека используется для создания всех окон, которые отображаются при использовании приложения, и содержимого этих окон, включая: тексты (label), поля (entry), кнопки (button), поле (canvas.create\_rectangle) и шашки (canvas.create\_circle).

**4. Сообщения**

«Чёрные (белые) не могут сделать, поэтому сейчас ходят белые (чёрные)».

«Игра окончена: чёрные (белые) победили, так как никто не может сделать ход».

«Игра окончена: так как у чёрных (белых) не осталось шашек на поле».

«Игра окончена: чёрные (белые) победили!».

«Игра окончена в ничью!».

**Текст программы**

import os

import tkinter as tk

from tkinter import font

from tkinter.font import Font

class Authorization(tk.Tk):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.title("Реверси by Иван Коцан")

self.canvas = tk.Canvas(self, width=400, height=550)

self.canvas.pack()

x = (self.winfo\_screenwidth() - 400) // 2

y = (self.winfo\_screenheight() - 550) // 2

self.geometry(f"400x550+{x}+{y}")

bold\_font = Font(family="Arial", size=12, weight="bold")

self.label = tk.Label(self,

text="Добро пожаловать в игру Реверси!\nДля начала игры пройдите процедуру\nавторизации/регистрации",

font=bold\_font)

self.label.config(justify=tk.CENTER)

self.label\_username = tk.Label(self, text="Введите имя пользователя:")

self.label\_password = tk.Label(self, text="Введите пароль:")

self.login = tk.Entry(self, width=30)

self.password = tk.Entry(self, width=30, show="•")

self.button\_register = tk.Button(self, text="Регистрация/авторизация", command=self.regist)

self.label.place(x=42, y=100)

self.login.place(x=112, y=300)

self.password.place(x=112, y=350)

self.label\_username.place(x=125, y=280)

self.label\_password.place(x=155, y=330)

self.button\_register.place(x=125, y=400)

def regist(self):

getlogin = self.login.get()

getpass = self.password.get()

if (len(getpass) == 0 or len(getlogin) == 0):

self.show\_info('Вы оставили пустым поле "Логин" и/или "Пароль"', "Повторить ввод")

else:

f\_reg = False

file\_path = 'accounts.txt'

if os.path.exists(file\_path):

file = open(file\_path, "r+")

text = file.read().split("\n")

for j in range(len(text) - 1):

if (text[j] == ""):

continue

str = text[j].split();

if str[0] == getlogin and str[1] == getpass:

f\_reg = True

break

elif str[0] == getlogin and str[1] != getpass:

f\_reg = "nopass"

file.close()

else:

with open(file\_path, 'w') as file:

file.write('')

if (f\_reg == "nopass"):

self.show\_info(getlogin + ", Вы указали неверный пароль.", "Повторить попытку")

else:

if not f\_reg:

file = open(file\_path, "r+")

file.seek(0, os.SEEK\_END)

file.write(getlogin + ' ' + getpass + '\n')

file.close()

self.show\_info("Добро пожаловать, " + getlogin + ". Вы успешно зарегистрировались.", "Начать игру!")

else:

self.show\_info("С возвращением, " + getlogin + ". Вы успешно авторизовались.", "Начать игру!")

self.label\_username.place\_forget()

self.label\_password.place\_forget()

self.login.place\_forget()

self.password.place\_forget()

self.button\_register.place\_forget()

def press(self, win\_t, auth): # функция нажатия кнопки в информационном окне

win\_t.grab\_release()

win\_t.destroy()

if auth == 1:

self.destroy()

game = ReversiGame()

game.mainloop()

def show\_info(self, text0, button0): #функция отображения информации при авторизации/регистрации

if button0 == "Начать игру!":

auth = 1

else:

auth = 0

info = tk.Toplevel(self)

info.title("Информация")

screen\_width = info.winfo\_screenwidth()

screen\_height = info.winfo\_screenheight()

x = (screen\_width // 2) - (500 // 2)

y = (screen\_height // 2) - (200 // 2)

info.geometry(f'500x200+{x}+{y}')

info.minsize(width=500, height=200)

info.maxsize(width=500, height=200)

info.protocol("WM\_DELETE\_WINDOW", lambda: self.press(info, auth))

bold\_font = Font(family="Arial", size=12, weight="bold")

label = tk.Label(info, text=text0, font=bold\_font)

label.pack()

label.place(relx=0.5, rely=0.2, anchor=tk.CENTER)

but = tk.Button(info, text=button0, command=lambda: self.press(info, auth))

but.place(x=200, y=125)

info.grab\_set()

BOARD\_SIZE = 8

EMPTY = 0

BLACK = 1

WHITE = 2

class ReversiGame(tk.Tk):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.kol\_black = 2 # кол-во чёрных шашек

self.kol\_white = 2 # кол-во белых шашек

self.symma = 4 # сумма всех шашек

self.win = "" # игрок, у которого больше шашек

self.yes = 0 # может ли текущий игрок сделать ход

self.mode\_ii = "lite" # уровень ИИ (none, lite, medium)

custom\_font = font.Font(family="Times New Roman", size=14)

custom\_font2 = font.Font(family="Times New Roman", size=12)

self.title("Реверси by Иван Коцан")

self.canvas = tk.Canvas(self, width=400, height=550)

self.canvas.pack()

self.board = [[EMPTY] \* BOARD\_SIZE for \_ in range(BOARD\_SIZE)]

self.board[3][3] = WHITE

self.board[3][4] = BLACK

self.board[4][3] = BLACK

self.board[4][4] = WHITE

self.current\_player = BLACK

self.infowhite = tk.Label(self, text="Белые - 2", font=custom\_font)

self.infowhite.place(x=50, y=405)

self.infoblack = tk.Label(self, text="Чёрные - 2", font=custom\_font)

self.infoblack.place(x=250, y=405)

self.odds = tk.Label(self, text="=", font=custom\_font)

self.odds.place(x=198, y=405)

self.thelp = tk.Label(self, text="Подсказка: чёрные делают ход", font=custom\_font2)

self.thelp.place(x=5, y=435)

self.button = tk.Button(self, text="Новая игра",command=self.new\_game)

self.button.place(x=150, y=500, width=100, height=30)

self.draw\_board()

self.canvas.bind("<Button-1>", self.on\_click)

x = (self.winfo\_screenwidth() - 400) // 2

y = (self.winfo\_screenheight() - 550) // 2

self.geometry(f"400x550+{x}+{y}")

def new\_game(self):

self.destroy()

self.\_\_init\_\_()

def draw\_board(self):

self.canvas.delete("all")

for row in range(BOARD\_SIZE):

for col in range(BOARD\_SIZE):

x1, y1 = col \* 50, row \* 50

x2, y2 = x1 + 50, y1 + 50

self.canvas.create\_rectangle(x1, y1, x2, y2, fill="green")

if self.board[row][col] == BLACK:

self.canvas.create\_oval(x1+5, y1+5, x2-5, y2-5, fill="black")

elif self.board[row][col] == WHITE:

self.canvas.create\_oval(x1+5, y1+5, x2-5, y2-5, fill="white")

def on\_click(self, event):

if event.y > 400:

return

row = event.y // 50

col = event.x // 50

if self.is\_valid\_move(row, col):

self.make\_move(row, col)

self.draw\_board()

def is\_valid\_move(self, row, col):

if self.board[row][col] != EMPTY:

return False

directions = [(0, 1), (1, 0), (0, -1), (-1, 0), (1, 1), (1, -1), (-1, 1), (-1, -1)]

for dx, dy in directions:

x, y = row + dx, col + dy

while 0 <= x < BOARD\_SIZE and 0 <= y < BOARD\_SIZE and self.board[x][y] == self.get\_opponent():

x += dx

y += dy

if 0 <= x < BOARD\_SIZE and 0 <= y < BOARD\_SIZE and self.board[x][y] == self.current\_player:

return True

return False

def get\_opponent(self):

return WHITE if self.current\_player == BLACK else BLACK

def check\_move(self):

self.yes = 0

for row in range(BOARD\_SIZE):

for col in range(BOARD\_SIZE):

if self.board[row][col] == EMPTY:

if self.is\_valid\_move(row, col):

self.yes = 1

self.movex, self.movey = row, col

break

return True if self.yes == 1 else False

def make\_ii\_move(self):

if self.mode\_ii == "lite":

if self.check\_move():

self.make\_move(self.movex, self.movey)

def cur\_player(self):

return "белые" if self.current\_player == WHITE else "чёрные"

def other\_player(self):

return "чёрные" if self.current\_player == WHITE else "белые"

def make\_move(self, row, col):

if self.current\_player == WHITE:

self.kol\_white = self.kol\_white + 1

else:

self.kol\_black = self.kol\_black + 1

self.board[row][col] = self.current\_player

directions = [(0, 1), (1, 0), (0, -1), (-1, 0), (1, 1), (1, -1), (-1, 1), (-1, -1)]

for dx, dy in directions:

x, y = row + dx, col + dy

while 0 <= x < BOARD\_SIZE and 0 <= y < BOARD\_SIZE and self.board[x][y] == self.get\_opponent():

x += dx

y += dy

if 0 <= x < BOARD\_SIZE and 0 <= y < BOARD\_SIZE and self.board[x][y] == self.current\_player:

while (x, y) != (row, col):

x -= dx

y -= dy

if self.board[x][y] != self.current\_player:

if self.current\_player == WHITE:

self.kol\_white += 1

self.kol\_black -= 1

else:

self.kol\_white -= 1

self.kol\_black += 1

self.board[x][y] = self.current\_player

self.current\_player = self.get\_opponent()

self.infowhite.config(text="Белые - " + str(self.kol\_white))

self.infoblack.config(text="Чёрные - " + str(self.kol\_black))

if self.kol\_white > self.kol\_black:

self.odds.config(text=">")

self.win = "Белые"

self.infowhite.config(fg="green")

self.infoblack.config(fg="black")

elif self.kol\_white < self.kol\_black:

self.odds.config(text="<")

self.win = "Чёрные"

self.infoblack.config(fg="green")

self.infowhite.config(fg="black")

else:

self.odds.config(text="=")

if self.win == "Чёрные":

self.infoblack.config(fg="black")

else:

self.infowhite.config(fg="black")

self.symma = self.kol\_white + self.kol\_black

if self.symma == 64:

if self.kol\_black != self.kol\_white:

self.thelp.config(text="Игра окончена: " + self.win + " победили!")

else:

self.thelp.config(text="Игра окончена в ничью!")

return

else:

if not self.check\_move():

self.thelp.config(text="Подсказка: " + self.cur\_player() + " не могут сделать ход,\nпоэтому сейчас ходят " + self.other\_player())

self.current\_player = self.get\_opponent()

if not self.check\_move(): #никто не может сделать ход

self.thelp.config(text="Игра окончена: " + self.win + " победили!\nТак как никто не мог сделать ход")

return

if self.current\_player == WHITE and self.mode\_ii != "none":

self.make\_ii\_move()

return

if self.kol\_white == 0:

self.thelp.config(text="Игра окончена: Чёрные победили,\nтак как у белых не осталось шашек на поле")

return

elif self.kol\_black == 0:

self.thelp.config(text="Игра окончена: Белые победили,\nтак как у чёрных не осталось шашек на поле")

if self.current\_player == WHITE:

self.thelp.config(text="Подсказка: белые делают ход")

if self.mode\_ii != "none":

self.make\_ii\_move()

else:

self.thelp.config(text="Подсказка: чёрные делают ход")

if \_\_name\_\_ == "\_\_main\_\_":

auth = Authorization()

auth.mainloop()