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:** «Color Lines»

**Инв. № подл.**

**Подп. и дата**

**Взам. инв. №**

**Инв. № дубл.**

**Подп. и дата**

Р.02069337. 22/2359-27 ТЗ-01

Листов: 39

Руководитель разработки:

к. т. н., доцент

Шишкин Вадим Викторинович

«28» декабря 2023 г.

Исполнитель:

студентка гр. ИСТбд-22

Рзянкина Виктория  
 Андреевна

«28» декабря 2023 г.

2023 г.

Содержание

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

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

Пояснительная записка...…………………………………………….......10

Руководство программиста……………………………….………….......21

Текст программы…..………………………………………………….......31

**Аннотация**

Данный документ представляет собой пояснительную записку на курсовую работу на тему «Color Lines». Документ содержит следующие разделы: техническое задание, пояснительная записка и руководство программиста, код программы; в нем излагается постановка задачи и описание реализуемой программы, ее назначение. Документ может быть использован в качестве инструкции для применения рассматриваемого программного средства.

**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

**Инв. № подл.**

**Подп. и дата**

**Взам. инв. №**

**Инв. № дубл.**

**Подп. и дата**

по дисциплине «Алгоритмы и структуры данных»

Тема: «Color Lines»

Р.02069337. 22/2359-27 ТЗ-01

Листов: 6

**Исполнитель**:

студентка гр. ИСТбд-22

Рзянкина Виктория Андреевна

«28» декабря 2023 г.

2023 г.

**Введение**

Курсовая работа представляет собой однопользовательское десктопное приложение, компьютерную игру «Color Lines»

Краткие правила игры:

* Компьютер выставляет три шарика случайных цветов на каждом ходу.
* Игрок может передвинуть любой шарик, если между начальной и конечной клетками есть недиагональный путь.
* Если собирается пять или более шариков одного цвета, они исчезают и игрок получает возможность сделать ещё одно перемещение.
* На экране показываются три цвета шариков, которые будут выброшены на поле на следующем ходу компьютера.

Функциональные возможности:

* Регистрация/авторизация пользователя
* Проверка правильности и отрисовка ходов пользователя и компьютера
* Графический интерфейс взаимодействия с пользователем

**1. Основания для разработки**

В качестве основания для разработки указывается учебный план направления 09.03.02 «Информационные системы и технологии».

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

**2.1. Функциональное назначение**

Требуется разработать однопользовательское десктопное приложение по игре в «Color Lines» с графическим интерфейсом в среде Windows.

**2.2. Требования к функциональным характеристикам**

Приложение должно соответствовать следующим правилам игры:

Игра происходит на квадратном поле в 9×9 клеток и представляет собой серию ходов. На каждом ходу сначала компьютер в случайных клетках выставляет три шарика случайных цветов, последних всего семь. Далее делает ход игрок, он может передвинуть любой шарик в другую свободную клетку, но при этом между начальной и конечной клетками должен существовать недиагональный путь из свободных клеток.

Если после перемещения получается так, что собирается пять или более шариков одного цвета в линию по горизонтали, вертикали или диагонали, то все такие шарики исчезают и игроку даётся возможность сделать ещё одно перемещение шарика. Если после перемещения линия не выстраивается, то ход заканчивается, и с появлением новых шариков начинается новый. Если при появлении новых шариков собирается линия, то она исчезает, игрок получает очки, но дополнительного перемещения не даётся. Игра продолжается до тех пор, пока всё поле не будет заполнено шариками и игрок не потеряет возможность сделать ход.

Цель игры состоит в наборе максимального количества очков. Счёт устроен таким образом, что игрок при удалении за одно перемещение большего числа шариков, чем пять, получает существенно больше очков. Во время игры на экране показывается три цвета шариков, которые будут выброшены на поле на следующем ходу.

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 прием, 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. Игровые изображения 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**

Указываются требования к работоспособности и способам восстановления при сбоях.

**2.4. Требования к информационной и программной совместимости**

Операционная система: Windows 10

Используемые библиотеки: tkinter, numpy, os

Язык: Python 3.9.1

Среда разработки: PyCharm Community Edition 2021.3.3

**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

ПРАВИЛЬНО:

1. Брать диски за внутреннее кольцо или за внешний край

2. Использовать специальный маркер для оптических дисков, не содержащий растворителя, для нанесения надписей на поверхность дисков

3. Избегать попадания грязи и других инородных предметов на поверхность дисков

4. Хранить диски в вертикальном положении (книжный способ) в пластиковых кейсах, предназначенных для CD и DVD

5. Помещать диск обратно в кейс сразу же после использования

6. Хранить диски всегда в упаковке для уменьшения влияния внешней среды

7. Открывать упаковку с дисками непосредственно перед записью

8. Хранить диски в темном, прохладном, сухом, проветриваемом помещении

9. Удалять грязь, инородные предметы, отпечатки пальцев, пятна и т.п. с поверхности дисков чистой хлопковой тканью легким движением от центра диска к внешнему краю

10. Использовать специальные чистящие средства для CD и DVD, изопропиленовый спирт или метанол для чистки поверхности дисков

11. Перед началом процесса записи проверять состояние поверхности диска

НЕ ПРАВИЛЬНО:

1. Прикасаться к поверхности диска

2. Гнуть диск

3. Размещать на диске наклейки

4. Хранить диски в горизонтальном положении длительное время (годами)

5. Открывать упаковку с диском задолго до начала записи

6. Хранить диски в очень жарком или очень влажном помещении

7. Хранить диски в условиях с большим колебанием температуры и влажности

8. Наносить надписи, пометки на пишущую поверхность диска (которую «читает» лазер)

9. Чистить диск круговыми движениями

ОСОБЕННО ДЛЯ CD НЕ ПРАВИЛЬНО:

1. Царапать нерабочую поверхность диска

2. Использовать ручки, карандаши или непредназначенные для CD фломастеры для нанесения надписей

3. Наносить надписи маркерами, содержащими растворители

4. Снимать или переклеивать наклейки

УСЛОВИЯ ХРАНЕНИЯ:

Температура – от 4°С до 20°С

Относительная влажность – от 20% до 50%

Температура 18°С и влажность 40% считаются подходящими для длительного хранения.

Более низкая температура и влажность считаются подходящими для продленного срока хранения.

2.6.3. Сроки хранения

Срок хранения – до июля 2023 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.

**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: «**Color Lines»

**Инв. № подл.**

**Подп. и дата**

**Взам. инв. №**

**Инв. № дубл.**

**Подп. и дата**

**Пояснительная записка**

Р.02069337. 22/2359-27 ПЗ-01

Листов: 11

**Исполнитель**:

студентка гр. ИСТбд-22

Рзянкина Виктория Андреевна

«28» декабря 2023 г.

2023 г.

**Введение**

Программа предоставляет собой игру «Color Lines».

**1. Проектная часть**

**1.1 Постановка задачи на разработку приложения**

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

**1.2 Математические методы**

В разрабатываемом приложении в качестве модели выступаtn случайно сгенерированное поля 9х9 для компьютера и визуальное поле для игрока. А также модель поиска пути для правила игры «Обязательное наличие пути».

Модель поля для компьютера:

['7', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ']

[' ', '6', ' ', ' ', ' ', ' ', '2', ' ', ' ']

[' ', ' ', '4', ' ', ' ', ' ', ' ', ' ', '5']

[' ', '5', ' ', ' ', '1', ' ', ' ', ' ', ' ']

[' ', ' ', ' ', '1', ' ', ' ', ' ', ' ', ' ']

[' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', '5']

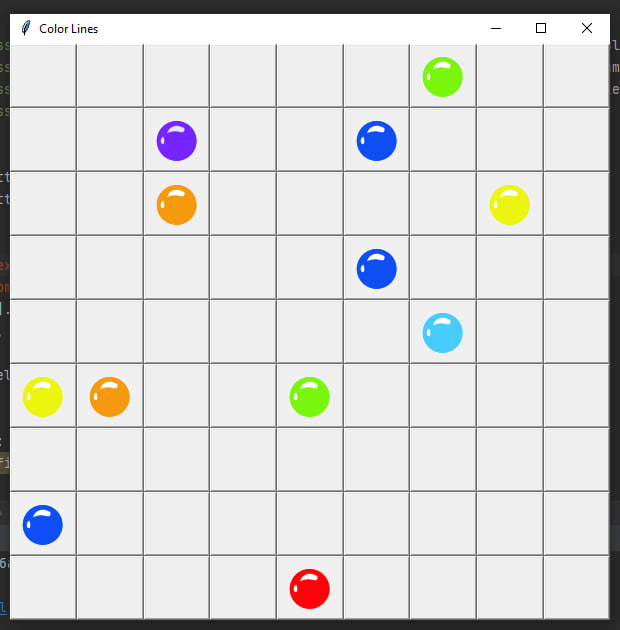
[' ', ' ', '7', ' ', ' ', ' ', ' ', ' ', '4']

[' ', ' ', ' ', ' ', '1', ' ', ' ', ' ', ' ']

['5', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ']

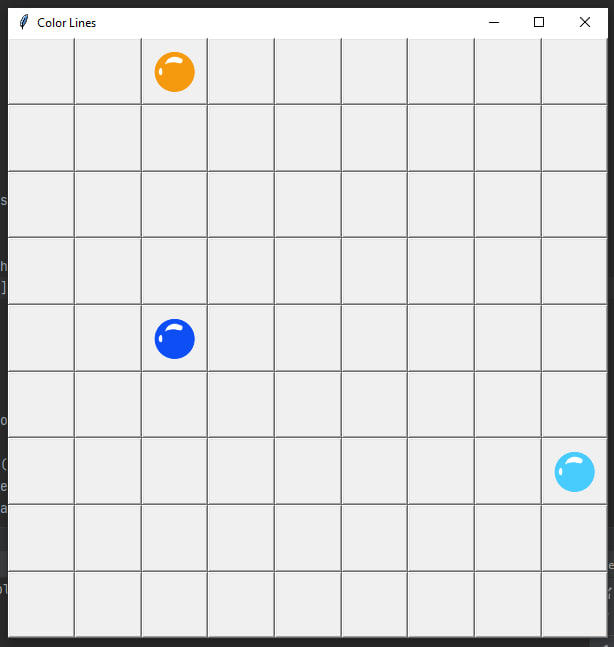
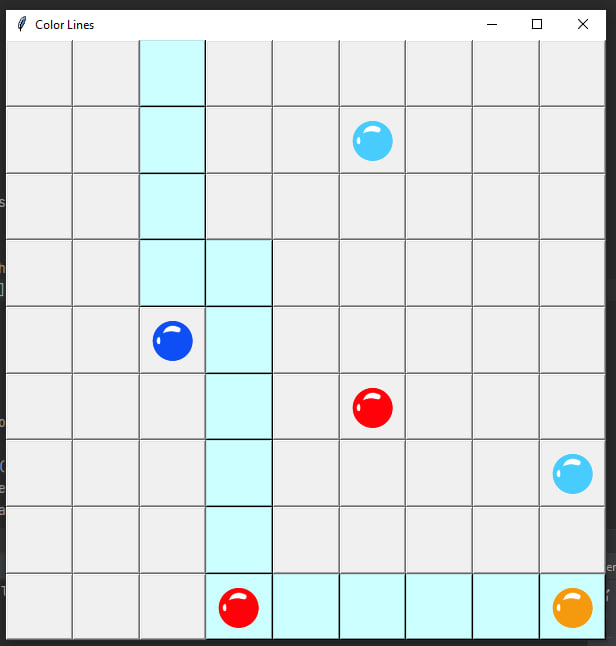
‘ ‘ – (пустые) клетки свободны, 1 – 7 – клетки с цветными шариками

Модель поля для игрока



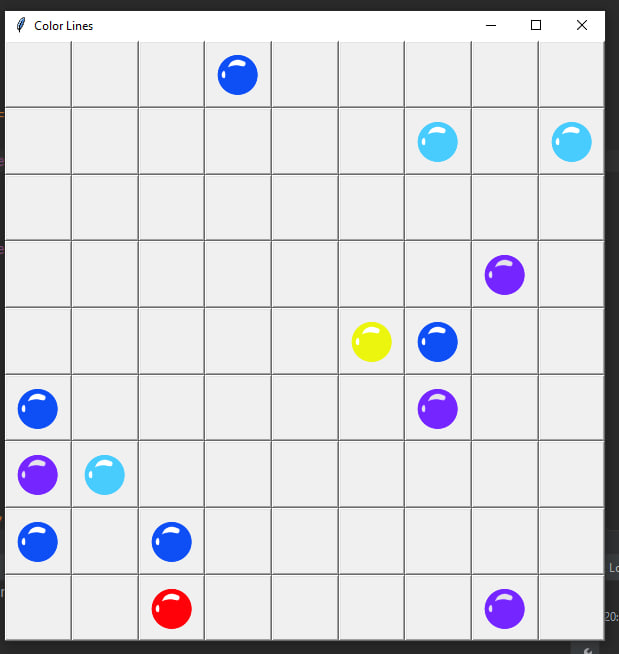
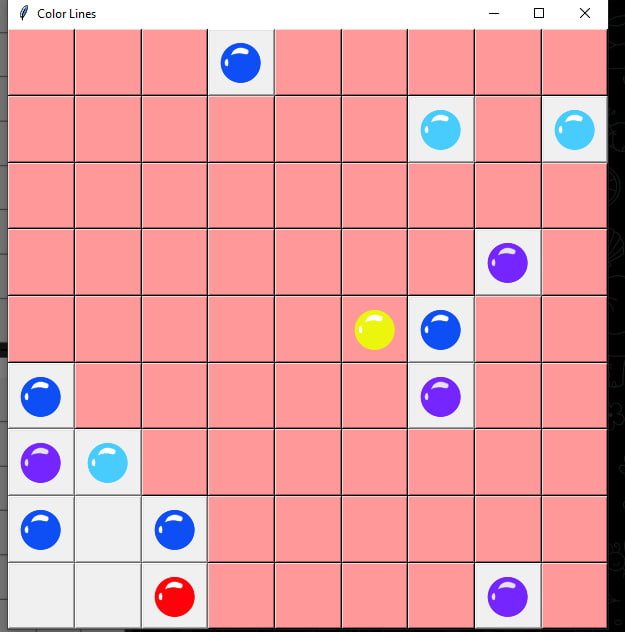
**Модель поиска пути**

Пример, когда шарик можно передвинуть. Голубым цветом показано как компьютер нашел наличие не диагональной дороги с точки «А» до точки «Б» (проверил правильность хода игрока), и передвинул шарик.



Пример, когда шарик нельзя передвинуть. Красным цветом показано, что компьютер проверил все возможные пути из «А» в «Б», но таких не оказалось, и шарик не был передвинут.



**1.3 Архитектура и алгоритмы**

1.3.1. Архитектура

Основной модуль

Отрисовка игрового поля

Авторизация

Игра

Обработчик событий

Вход/Регистрация

Ход компьютера

Подсчет очков

Ход игрока

Проверка правильности хода игрока

Определение конца игры

1.3.2. Алгоритм создания игрового поля:

Начало

Создание игрового поля

Объявление шариков компьютером

Объявление шариков компьютером

да

Ход игрока, передвижение шариков

Ход правильный?

нет

да

да

Счёт изменился?

нет

На поле осталось место?

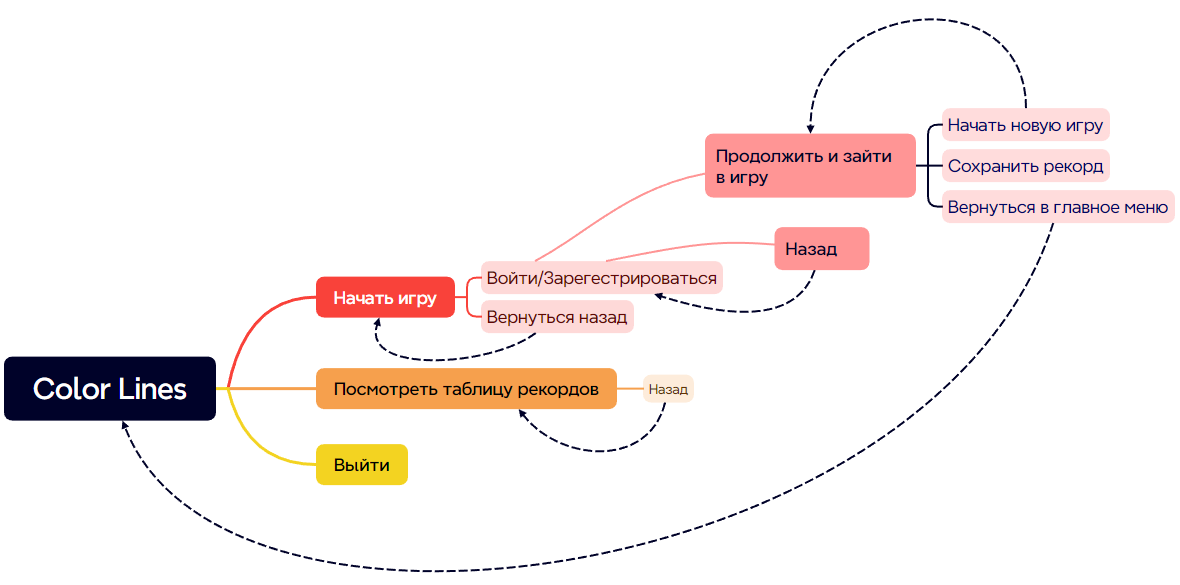
да

нет

Конец игры

**1.4 Тестирование**

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

Целью тестирования является проверка соответствия ПО предъявляемым требованиям, а также выявление возможных багов. По результатам тести-

рования следует исправление выявленных багов.

1.4.3 Методика тестирования

Тестирование проводилось с использованием следующих методов:

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

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

1.4.4 Проведенные тесты

4.1) Статическое тестирование

Количество обнаруженных и исправленных ошибок в документации: 3.

Количество обнаруженных и исправленных ошибок в программном

коде: 1.

В ходе тестирования были проведены следующие тесты:

4.2) Ручное тестирование

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

**TK1.** Проверка отрисовки игрового поля

Шаги:

1. Запустить приложение

2. В появившемся окне нажать кнопку ‘Начать игру’, затем ‘Регистрация’.

3.Заполнить поля и нажать ‘Продолжить’

Ожидаемый результат:

Должно открыться окно с отрисованным игровым полем и виджетами

Фактический результат:

Открылось окно с отрисованным игровым полем и виджетами

**ТК2.** Проверка хода

1. Запустить приложение

2. В появившемся окне нажать кнопку ‘Начать игру’, затем ‘Регистрация’.

3. Заполнить поля и нажать ‘Продолжить’

4. Проверить возможность передвижения шариков, а также верное выполнение правила «наличия недиагонального пути от начальной клетки до конечной.

Ожидаемый результат:

Передвижение шариков возможно, проверка наличия пути осуществляется.

Фактический результат:

Передвижение шариков возможно, проверка наличия пути осуществляется.

**ТК3.** Проверка окончания игры.

1. Запустить приложение

2. В появившемся окне нажать кнопку ‘Начать игру’, затем ‘Регистрация’.

3. Заполнить поля и нажать ‘Продолжить’

4. Дойти до стадии игры, когда на поле не остается места для новых шариков.

Ожидаемый результат:

Появление предупреждения об окончании игры, открытие окна с виджетами для выхода из игры или перезапуском игры.

Фактический результат:

При отсутствии свободных клеток, возможность передвигать шарики с последним «фантомным»(шарик, который компьютер хочет поставить на следующем ходу) постоянно повторяется. Игра не оканчивается вовсе.

**ТК4.** Проверка правильности отрисовки всех виджетов и изображений.

1. Запустить приложение.

2. В появившихся окнах поочередно нажимать все кнопки.

3. Внимательно просмотреть наличие изображений и виджетов.

Ожидаемый результат:

Корректная работа всех изображений и виджетов.

Фактический результат:

Корректная работа всех изображений и виджетов.

**ТК5**. Проверка появления шариков.

1. Запустить приложение

2. В появившемся окне нажать кнопку ‘Начать игру’, затем ‘Регистрация’.

3. Заполнить поля и нажать ‘Продолжить’

4. Проверить, всегда ли ставятся ровно три шарика во время хода компьютера.

Ожидаемый результат:

Шарики появляются без сбоев.

Фактический результат:

Шарики появляются без сбоев.

**ТК6.** Проверка корректности работы всех кнопок в соответствии с архитектурой.

1. Запустить приложение
2. В появившихся окнах поочередно нажимать все кнопки.
3. Проверить верную работу кнопок.

Ожидаемый результат:

Корректная работа кнопок.

Фактический результат:

Корректная работа кнопок.

Чек-лист для полей ввода

|  |  |  |
| --- | --- | --- |
| Проверка | Ожидаемый результат | Фактический результат |
| Проверка отрисовки игрового поля | Должно открыться окно с отрисованным игровым полем и виджетами | Открылось окно с отрисованным игровым полем и виджетами |
| Проверка хода | Передвижение шариков возможно, проверка наличия пути осуществляется. | Передвижение шариков возможно, проверка наличия пути осуществляется. |
| Проверка окончания игры | Появление предупреждения об окончании игры, открытие окна с виджетами для выхода из игры или перезапуском игры. | При отсутствии свободных клеток, возможность передвигать шарики с последним «фантомным»(шарик, который компьютер хочет поставить на следующем ходу) постоянно повторяется. Игра не оканчивается вовсе. |
| Проверка правильности отрисовки всех виджетов и изображений. | Корректная работа всех изображений и виджетов. | Корректная работа всех изображений и виджетов. |
| Проверка на появление 3 шариков | Шарики появляются без сбоев. | Шарики появляются без сбоев. |
| Проверка корректности работы всех кнопок в соответствии с архитектурой. | Корректная работа кнопок. | Корректная работа кнопок. |

1.4.5 Выводы

На основе проведенных тестов сделаны следующие выводы:

– Программа успешно прошла все тесты и работает корректно.

– Обнаружены и исправлены следующие дефекты: некорректная работа завершения игры.

**2. Источники, использованные при разработке**

1. Модуль datetime в Python – функции и методы на примерах [Электронный ресурс] - Режим доступа: для всех пользователей URL: <https://pythonru.com/primery/kak-ispolzovat-modul-datetime-vpython?ysclid=lqi36ynjaz672975391> (дата обращения: 26.10.23).

2. Руководство по программированию на Tkinter и Python [Электронный ресурс]. – Режим доступа: для всех пользователей. – URL: <https://metanit.com/python/tkinter/?ysclid=lq9y8691w1850842283> (дата обращения: 9.11.23).

3. Документация Python 3.12.1. [Graphical User Interfaces with Tk](https://docs.python.org/3/library/tk.html) [Электронный ресурс]. – Режим доступа: для всех пользователей. – URL: https://docs.python.org/3/library/tk.html (дата обращения: 05.11.23).

4. Шишкин, В.В. Разработка логических компьютерных игр с графическим интерфейсом в среде Python [Электронный ресурс] / В.В. Шишкин, Д.С. Афонин. – Ульяновск: УлГТУ, 2023. – 89 с. – Режим доступа: для всех пользователей. – URL: <http://lib.ulstu.ru/venec/disk/2023/112.pdf> (дата обращения: 17.12.23).

5. Материал из Википедии – свободной энциклопедии; оригинальные правила игры[Электронный ресурс]. – Режим доступа: для всех пользователей. – URL: <https://ru.wikipedia.org/wiki/Color_Lines> (дата обращения 24.10.23)

**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:** Color Lines

**Инв. № подл.**

**Подп. и дата**

**Взам. инв. №**

**Инв. № дубл.**

**Подп. и дата**

**Руководство программиста**

Р.02069337. 22/2359-27 РП-<01>

Листов: 19

Исполнитель:

студентка гр. ИСТбд-22

Рзянкина Виктория Андреевна

«28» декабря 2023 г.

2023 г.

**1. Назначение и условия применения программы**

**1.1 Назначение и функции, выполняемые приложением**

Программа предназначена для игры в компьютерную игру «Color Lines»

Краткие правила игры:

* Компьютер выставляет три шарика случайных цветов на каждом ходу.
* Игрок может передвинуть любой шарик, если между начальной и конечной клетками есть недиагональный путь.
* Если собирается пять или более шариков одного цвета, они исчезают и игрок получает возможность сделать ещё одно перемещение.
* На экране показываются три цвета шариков, которые будут выброшены на поле на следующем ходу компьютера.

Цель игры – набрать наибольшее количество очков.

**1.2 Условия, необходимые для использования приложения**

Необходимы следующие библиотеки для использования приложения: tkinter, pygame, PIL, hashlib, random.

Требования к операционной системе: Windows 10 и выше.

Требования к платформе: любая современная платформа с поддержкой Python версии 3.10.

**2. Характеристики программы**

**2.1 Характеристики приложения**

Количество значимых строк кода: 415.

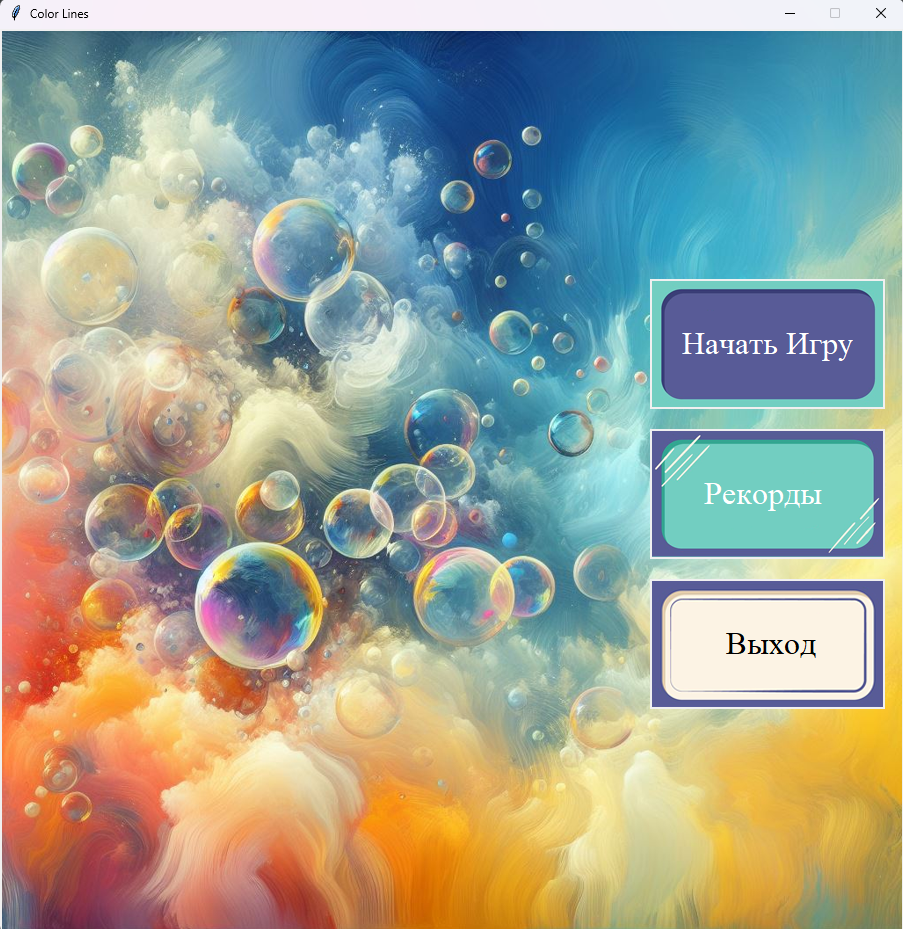
Количество алгоритмов: 32.

Используемые библиотеки:

1. tkinter – библиотека для разработки графического интерфейса; в приложении используется для создания графического окна с виджетами;
2. pygame – Набор модулей языка программирования Python, предназначенный для написания компьютерных игр и мультимедиа-приложений; в приложении используется для воспроизведения музыки в игре;
3. PIL – библиотека языка Python (версии 2), предназначенная для работы с растровой графикой; в приложении библиотека используется для использования изображений библиотекой tkinter;
4. hashlib - Модуль реализует общий интерфейс для множества различных безопасных алгоритмов хеширования и дайджеста сообщений; в программе используется для шифрования пароля посредством хэширующего алгоритма;
5. random – Этот модуль реализует генераторы псевдослучайных чисел под различные потребности.

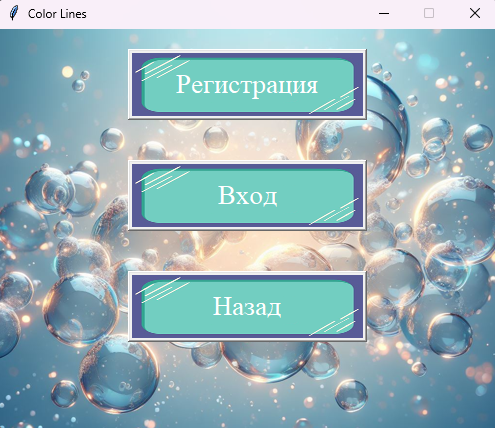
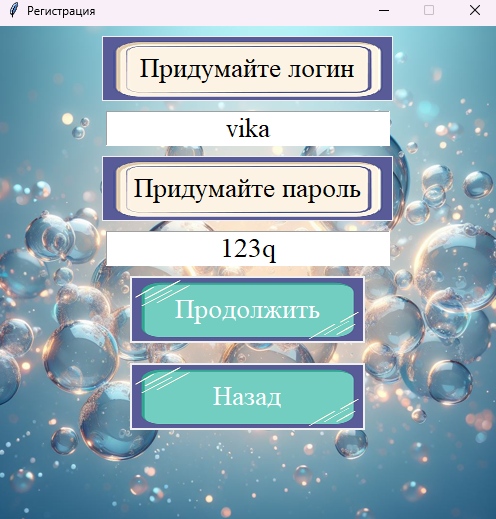
Описание приложения:

При запуске приложения появляется окно главного меню.

****

Пользователю предоставляется на выбор три действия: при нажатии на кнопку “Начать игру” откроется окно с выбором регистрации или авторизации, а также кнопка возвращения назад в главное меню; при нажатии на кнопку “Рекорды” пользователь сможет увидеть таблицу собственных рекордов, сохраненных ранее; при нажатии на кнопку “Выход” приложение завершит свою работу

****

**** ****

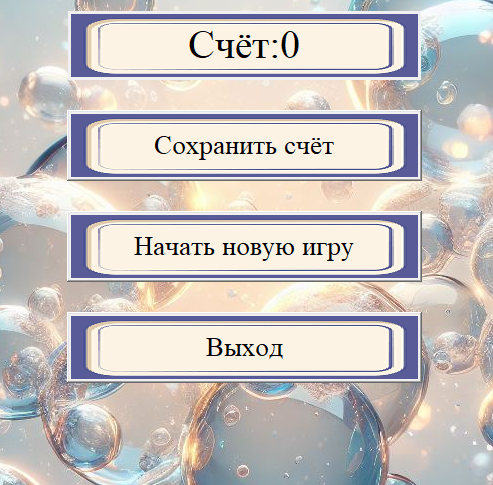
После авторизации/регистрации откроется окно с игрой, в котором отрисовано поле со случайно расставленными 3-мя шариками, и 3 «фантомными» шариками, которые будут поставлены компьютеров в следующем ходу.



Справа от игрового поля пользователь видит свой счёт и несколько кнопок:

* Сохранить рекорд
* Начать новую игру
* Вернуться в главное меню

При проигрыше поле с игрой закрывается и открывается новое окно, предлагающее сохранить рекорд, начать новую игру или выйти.



**2.2 Особенности реализации приложения**

Использованные в программе структуры данных:

1. строки – главная структура данных в приложении, поскольку работа с файлами осуществляется на основе строкового представления данных; хранят содержимое файлов, их названия, кодировки и др;
2. списки – структура данных, применяемая для хранения путей к выбранным файлам, соответствующие им кодировки для корректной обработки;
3. целые числа – структура данных, необходимая для нумерации строк, записываемых в новый файл, а также нумерации выбранных файлов при их удалении в автоматическом режиме работы;
4. булевые значения – true/false – структура данных, используемая для проверки достижения конца файла, заполнения пользователем всех необходимых полей, наличия ошибок в файле.

**3. Обращение к программе**

Функции:

Class Main

1. Main\_menu – Открытие и отрисовка главного меню;
2. table\_records – Открытие и отрисовка таблицы рекордов;
3. back\_record – функция для кнопки возвращения в главное меню из таблицы рекордов.

Class Auth

1. \_\_init\_\_ - Открытие и отрисовка поля выбора авторизации/регистрации;
2. back\_command – функция для кнопки возвращение в меню выбора/главное;
3. window\_entry – подготовка полей ввода логина и пароля;
4. select\_auth – Открытие и отрисовка поля авторизации/авторизации;
5. register – вызов функции ввода(window\_entry) для регистрации;
6. log\_in – вызов функции ввода(window\_entry) для авторизации;
7. open\_txt – открытие файла для проверки данных об авторизации, а также проверка на наличие ошибок в полях ввода; запуск функции игры;
8. write\_txt - открытие файла для записи данных об регистрации, а также проверка на наличие ошибок в полях ввода; запуск функции игры.

Class Game

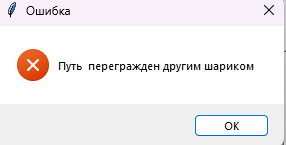
1. \_\_init\_\_ - подготовка переменных для дальнейшей работы;
2. start\_game – открытие и отрисовка окна и поля игры; отрисовка основных виджетов;
3. back\_command – функция для кнопок выхода в главное меню;
4. new\_game – для запуска новой игры;
5. add\_color\_ball – отрисовка визуального шарика;
6. add\_ghost\_ball – отрисовка визуального «фантомный» шарика;
7. stage\_choice – определение стадии игры;
8. computer\_spawn\_ball – компьютер ставит шарик в соответствии с выбором ячеек и числом, обозначающим цвет;
9. computer\_predict\_space\_ball – выбор компьютером ячеек для установки на следующем ходу; проверка на заполненность поля; Открытие и отрисовка окна конца игры;
10. player\_choice\_ball – выбор игроком шарика, сохранение его координат;
11. player\_place\_ball –ставится шарик, либо выбирается новый;
12. validate\_coord – проверка на то что координаты находятся в пределах поля;
13. near\_color\_check – проверка ближайших координат от поставленного игроков шарика на одинаковый цвет;
14. line\_calculate – проверка последующих координат на тот же подряд идущий цвет шарика;
15. delete\_ball - удаление шарика;
16. score\_calculate – подсчет очков;
17. pathfinding – поиск наличия пути от начальной координаты до конечной;
18. get\_adjacent-nodes – получение ближайших координат;
19. choose\_node – выбор лучшего пути, для ускорения поиска;
20. write\_score – открытие файла для сохранения счета игрока.

Используемые библиотеки:

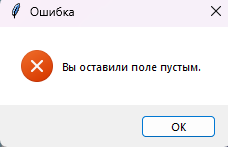
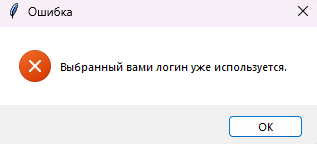
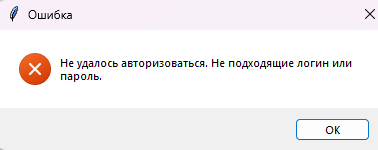
1. tkinter – библиотека для разработки графического интерфейса; в приложении используется для создания графического окна с виджетами;
2. pygame – Набор модулей языка программирования Python, предназначенный для написания компьютерных игр и мультимедиа-приложений; в приложении используется для воспроизведения музыки в игре;
3. PIL – библиотека языка Python (версии 2), предназначенная для работы с растровой графикой; в приложении библиотека используется для использования изображений библиотекой tkinter;
4. hashlib - Модуль реализует общий интерфейс для множества различных безопасных алгоритмов хеширования и дайджеста сообщений; в программе используется для шифрования пароля посредством хэширующего алгоритма;
5. random – Этот модуль реализует генераторы псевдослучайных чисел под различные потребности.

**4. Сообщения**

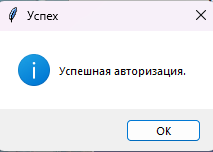
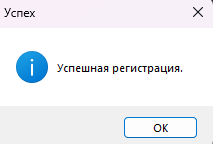
При попытке сходить не в соответствии с правилами игры, пользователь видит сообщение:



Если во время регистрации пользователь ошибается, от видит предупреждения:

Если же все прошло успешно:



**Текст программы:**

from pygame import mixer  
from tkinter import \*  
from tkinter import messagebox  
from PIL import ImageTk, Image  
import hashlib  
import random

mixer.init()  
mixer.music.load('assets/MenuSound.mp3')  
mixer.music.play(loops=-1)  
mixer.music.set\_volume(0.1)  
  
GAME\_OVER = 'game\_over'  
SELECT = 'choice'  
MOVE = 'move'  
  
img = Image.open('assets/menuu.jpeg')  
auth\_bg = Image.open('assets/menu (2).jpeg')  
  
class Main():  
 def \_\_init\_\_(self):  
 self.root = Tk()  
 self.root.title("Color Lines")  
 self.Main\_menu()  
  
 def Main\_menu(self):  
 imag = img.resize((900, 900))  
 background = ImageTk.PhotoImage(imag, master = self.root)  
 bg = Label( self.root, image=background)  
 bg.pack()  
 self.root.eval('tk::PlaceWindow . center')  
 self.root.resizable(False,False)  
 button\_image\_start = PhotoImage(file='assets/button111.png')  
 button\_image\_exit = PhotoImage(file='assets/button333.png')  
 button\_image\_score = PhotoImage(file='assets/button222.png')  
  
 button\_play = Button( self.root, text = 'Начать Игру', fg='#ffffff', font='Times 24', border="0", image = button\_image\_start, compound='center', command= lambda: Auth( self.root))  
  
 button\_score = Button( self.root, text = 'Рекорды ', fg='#ffffff', font='Times 24', border="0", image = button\_image\_score, compound='center', command=self.table\_records)  
 button\_exit = Button( self.root, text = ' Выход', font='Times 24', border="0", image = button\_image\_exit, compound='center', command = self.root.destroy )  
  
 button\_play.place(x=650, y=250)  
  
 button\_score.place(x=650, y=400)  
 button\_exit.place(x=650, y=550)  
 self.root.mainloop()  
  
 def table\_records(self):  
 self.root.destroy()  
 records\_window = Tk()  
 button\_image\_records = PhotoImage(master= records\_window, file='assets/button333.png').subsample(1,2)  
 button\_image\_back = PhotoImage(master= records\_window, file='assets/button222.png').subsample(1,2)  
 records\_window.title("Color Lines")  
 records\_window.geometry('400x900')  
 records\_window.resizable(False,False)  
 background = ImageTk.PhotoImage(auth\_bg, master = records\_window)  
 bg = Label(records\_window, image=background)  
 bg.place(x=0, y=0, relwidth=1, relheight=1)  
 count = 0  
 with open('ScoreFile.txt', 'r') as f:  
 lines = f.readlines()  
 for line in lines:  
 if line == '':  
 break  
 if count == 10:  
 break  
 else:  
 readline\_score = line[:-1].split(':', 2)  
 Label(records\_window, text = readline\_score, font='Times 24', border="0", image = button\_image\_records, compound='center').pack(pady= 3)  
 count += 1  
 Button(records\_window, text = 'Назад',font='Times 24', fg='#ffffff', border="0", image = button\_image\_back, compound='center', command= lambda:self.back\_record(records\_window)).place(x=80, y=800)  
 records\_window.eval('tk::PlaceWindow . center')  
 records\_window.mainloop()  
 def back\_record(self,records\_window):  
 records\_window.destroy()  
 Main()  
  
class Auth:  
 def \_\_init\_\_(self, menu):  
 menu.destroy()  
 open('File.txt', 'a').close()  
 self.window = Tk()  
 self.window.title("Color Lines")  
 window = self.window  
 window.geometry('500x400')  
 button\_image\_auth = PhotoImage(file='assets/button222.png').subsample(1,2)  
 self.imag = auth\_bg.resize((500, 500))  
 background = ImageTk.PhotoImage(self.imag)  
 bg = Label(window, image=background)  
 bg.place(x=0, y=0, relwidth=1, relheight=1)  
 self.window.resizable(False, False)  
 Button(window, font='Times 20', fg='#ffffff', compound='center', text="Регистрация", image =button\_image\_auth, command=self.register).pack(pady=20)  
 Button(window, font='Times 20', fg='#ffffff', text="Вход", compound='center', image =button\_image\_auth, command=self.log\_in).pack(pady=20)  
 but = Button(window, text="Назад", font='Times 20', fg='#ffffff', compound='center', image =button\_image\_auth, command=lambda: self.back\_command(False))  
 but.pack(pady=20)  
 window.eval('tk::PlaceWindow . center')  
 self.window.mainloop()  
  
 def back\_command(self, key):  
 if key == False:  
 self.window.destroy()  
 Main()  
 else:  
 Auth(self.auth\_window)  
  
 def window\_entry(self, description, img):  
 Label(self.auth\_window, font='Times 20', border="0", text=description, image=img, compound='center').pack(pady=10)  
 entry = Entry(self.auth\_window, font='Times 20', width= 20, justify='center')  
 entry.pack()  
 return entry  
  
 def select\_auth(self, title, log, passw, command):  
 self.window.destroy()  
 self.auth\_window = Tk()  
 self.auth\_window.title(title)  
 wind = self.auth\_window  
 wind.geometry('500x500')  
 button\_image\_auth = PhotoImage(file='assets/button222.png').subsample(1, 2)  
 background = ImageTk.PhotoImage(self.imag, master= wind)  
 bg = Label(self.auth\_window, image=background)  
 bg.place(x=0, y=0, relwidth=1, relheight=1)  
 self.auth\_window.resizable(False, False)  
 img = PhotoImage(file='assets/button333.png').zoom(5,1).subsample(4,2)  
 self.login\_user = self.window\_entry(log,img)  
 self.password\_user = self.window\_entry(passw,img)  
 Button(wind, font='Times 20', border="0", compound='center', fg='#ffffff', text="Продолжить", image =button\_image\_auth, command=command).pack(pady=10)  
 Button(wind, font='Times 20', border="0", compound='center', fg='#ffffff', text ='Назад', image =button\_image\_auth, command=lambda: self.back\_command(True)).pack(pady=10)  
 wind.eval('tk::PlaceWindow . center')  
 self.auth\_window.mainloop()  
 def register(self):  
  
 self.select\_auth('Регистрация', 'Придумайте логин', 'Придумайте пароль', self.write\_txt)  
 self.auth\_window.mainloop()  
  
 def log\_in(self):  
  
 self.select\_auth('Вход', 'Введите логин', 'Введите пароль', self.open\_txt)  
 self.auth\_window.mainloop()  
 def open\_txt(self):  
 str\_user = self.login\_user.get()  
 not\_found = True  
 with open('File.txt', 'r') as f:  
 read = f.readlines()  
 password\_sha = hashlib.sha1(str.encode(self.password\_user.get())).hexdigest()  
 for i in range(1, len(read), 2):  
 if str\_user == read[i].rstrip('\n') and password\_sha == read[i + 1].rstrip('\n'):  
 not\_found = False  
 messagebox.showinfo('Успех', 'Успешная авторизация.')  
 self.auth\_window.destroy()  
 Game(str\_user).start\_game()  
 break  
 if str\_user == '' or self.password\_user.get() == '':  
 messagebox.showerror('Ошибка','Вы оставили поле пустым.')  
 return  
 if not\_found:  
 messagebox.showerror('Ошибка','Не удалось авторизоваться. Не подходящие логин или пароль.')  
  
 def write\_txt(self):  
 str\_user = self.login\_user.get()  
 with open('File.txt', 'r') as f:  
 read = f.readlines()  
 for i in range(1, len(read), 2):  
 if str\_user == read[i].rstrip('\n'):  
 messagebox.showerror('Ошибка','Выбранный вами логин уже используется.')  
 return  
 if self.login\_user.get() == '' or self.password\_user.get() == '':  
 messagebox.showerror('Ошибка','Вы оставили поле пустым.')  
 return  
 with open('File.txt', 'a') as f:  
 f.write(  
 '\n' + str\_user + '\n' + hashlib.sha1(str.encode(self.password\_user.get())).hexdigest())  
 f.close()  
 self.auth\_window.destroy()  
 messagebox.showinfo('Успех', 'Успешная регистрация.')  
 Game(str\_user).start\_game()  
  
  
class Game:  
  
 def \_\_init\_\_(self,login\_user):  
  
 open('ScoreFile.txt', 'a').close()  
 self.login\_user = login\_user  
 self.turns = 0  
 self.colors = ['1', '2', '3', '4', '5', '6', '7']  
 self.ghost\_colors = ['-1', '-2', '-3', '-4', '-5', '-6', '-7']  
 self.current\_stage = SELECT  
 self.buttons = [[], [], [], [], [], [], [], [], []] *# кнопки* self.score = 0  
 self.future\_balls = []  
  
 def start\_game(self):  
 self.game\_window = Tk()  
 self.game\_window.title("Color Lines")  
 self.game\_window.geometry("1200x865")  
 background = ImageTk.PhotoImage(img.resize((1200, 1000)))  
 self.game\_window.resizable(False,False)  
 bg = Label(self.game\_window, image=background)  
 bg.place(x=0, y=0, relwidth=1, relheight=1)  
  
 self.mini\_image = [  
 PhotoImage(file="assets/red.png").subsample(2, 2), PhotoImage(file="assets/yellow.png").subsample(2, 2),  
 PhotoImage(file="assets/orange.png").subsample(2, 2), PhotoImage(file="assets/green.png").subsample(2, 2),  
 PhotoImage(file="assets/blue.png").subsample(2, 2), PhotoImage(file="assets/quan.png").subsample(2, 2),  
 PhotoImage(file="assets/purple.png").subsample(2, 2)  
 ]  
 self.image = [  
 PhotoImage(file="assets/red.png"), PhotoImage(file="assets/yellow.png"),  
 PhotoImage(file="assets/orange.png"), PhotoImage(file="assets/green.png"),  
 PhotoImage(file="assets/blue.png"), PhotoImage(file="assets/quan.png"),  
 PhotoImage(file="assets/purple.png")  
 ]  
 self.empty =PhotoImage(file='assets/empty.png').subsample(2, 2)  
 button\_image\_gamewindow = PhotoImage(file='assets/button333.png').zoom(5,1).subsample(4,2)  
 button\_image\_score= PhotoImage(file='assets/button333.png').zoom(1,1).subsample(1,2)  
 for r in range(9):  
 for c in range(9):  
  
 btn = Button( text=' ', image= self.empty, width=90, height=90,  
 command=lambda row=r, column=c: self.stage\_choice(row, column))  
  
 self.buttons[r].append(btn)  
 btn.grid(row=r, column=c, sticky="nsew")  
 self.computer\_predict\_space\_ball()  
 self.computer\_predict\_space\_ball()  
 self.score\_label = Label(self.game\_window, text=f'Счёт:{self.score}', font='Times 30', image=button\_image\_score, compound='center')  
 self.score\_label.place(x= 900,y= 20)  
 self.save\_button = Button(self.game\_window, border="0", text ='Сохранить рекорд', image= button\_image\_gamewindow, font='Times 20', compound='center', command=self.write\_score)  
 self.save\_button.place(x = 900, y = 100)  
 self.new\_game\_button = Button(self.game\_window, border="0", text='Начать новую игру',image= button\_image\_gamewindow, font='Times 20', compound='center', command=self.new\_game)  
 self.new\_game\_button.place(x=900, y=700)  
 self.button\_back = Button(self.game\_window, text="В главное меню", compound='center', image=button\_image\_gamewindow,font='Times 20', command=lambda: self.back\_command(False))  
 self.button\_back.place(x=900, y= 780)  
 self.game\_window.eval('tk::PlaceWindow . center')  
 self.game\_window.mainloop()  
 def back\_command(self, key):  
 if key == False:  
 self.game\_window.destroy()  
 Main()  
 else:  
 Auth(self.auth\_window)  
  
 def new\_game(self,end\_window = None):  
 if end\_window != None:  
 end\_window.destroy()  
 else:  
 self.game\_window.destroy()  
 Game(self.login\_user).start\_game()  
  
 def add\_color\_ball(self, x, y):  
 if self.buttons[x][y]['text'] == ' ':  
 return  
 self.buttons[x][y].config(width=90, height=90, image=self.image[int(self.buttons[x][y]['text']) - 1])  
 def add\_ghost\_ball(self, x, y, color):  
 if color == ' ': return  
 self.buttons[x][y].config(width=90, height=90, image=self.mini\_image[int(color) - 1])  
  
 def stage\_choice(self, x, y):  
 if self.current\_stage == SELECT:  
 self.player\_choice\_ball(x, y)  
 elif self.current\_stage == MOVE:  
 self.player\_place\_ball(x, y)  
  
 def computer\_spawn\_ball(self):  
 empty = []  
 for x, y, color in self.future\_balls:  
 if self.buttons[x][y]['text'] != ' ':  
 for x in range(8):  
 for y in range(8):  
 if self.buttons[x][y]['text'] == ' ':  
 empty.append([x,y])  
 x,y = random.choice(empty)  
 color = random.choice(self.colors)  
 self.buttons[x][y]['text'] = color  
 self.near\_color\_check(x, y)  
 self.add\_color\_ball(x, y)  
 self.future\_balls.clear()  
  
 def computer\_predict\_space\_ball(self):  
 self.computer\_spawn\_ball()  
 empty\_cells = []  
 for x in range(9):  
 for y in range(9):  
 if self.buttons[x][y]['text'] == ' ':  
 empty\_cells.append([x, y])  
 if len(empty\_cells) == 1:  
 end\_window = Tk()  
 self.game\_window.destroy()  
 end\_window.title('Color Lines')  
 end\_window.geometry('500x500')  
 button\_img = PhotoImage(master= end\_window, file='assets/button333.png').zoom(3,1).subsample(2,2)  
 background = ImageTk.PhotoImage(auth\_bg, master=end\_window)  
 bg = Label(end\_window, image=background)  
 bg.place(x=0, y=0, relwidth=1, relheight=1)  
 end\_window.resizable(False, False)  
 Label(end\_window, text=f'Счёт:{self.score}', font='Times 30',image=button\_img, compound='center').pack(pady=15)  
 Button(end\_window, text='Сохранить счёт', compound='center', image=button\_img, font='Times 20', command=self.write\_score).pack(pady=15)  
 Button(end\_window,compound='center', image=button\_img ,text='Начать новую игру', font='Times 20',command= lambda:self.new\_game(end\_window)).pack(pady=15)  
 Button(end\_window, text='Выход', font='Times 20', compound='center', image=button\_img,command=end\_window.destroy).pack(pady=15)  
 end\_window.eval('tk::PlaceWindow . center')  
 end\_window.mainloop()  
 return  
 max\_places = min(len(empty\_cells), 3)  
 while max\_places != 0:  
 select = random.randint(0, len(empty\_cells) - 1)  
 x, y = empty\_cells.pop(select)  
 if self.buttons[x][y]['text'] == ' ':  
 color = random.choice(self.colors)  
 self.future\_balls.append([x, y, color])  
 self.add\_ghost\_ball(x, y, color)  
 max\_places -= 1  
  
 def player\_choice\_ball(self, x, y):  
 if self.buttons[x][y]['text'] != ' ':  
 self.save\_x = x  
 self.save\_y = y  
 self.buttons[x][y].config(bg='#98FB98')  
 self.current\_stage = MOVE  
  
 def player\_place\_ball(self, x, y):  
 if self.buttons[x][y]['text'] == ' ':  
 if not self.pathfinding(x, y):  
 messagebox.showerror('Ошибка','Путь перегражден другим шариком')  
 return  
 self.buttons[self.save\_x][self.save\_y].config(bg='SystemButtonFace')  
 self.buttons[x][y]['text'] = self.buttons[self.save\_x][self.save\_y]['text']  
 self.current\_stage = SELECT  
 self.add\_color\_ball(x, y)  
 self.delete\_ball(self.save\_x, self.save\_y)  
 previous\_score = self.score  
 self.near\_color\_check(x, y)  
 if self.score == previous\_score :  
 self.computer\_predict\_space\_ball()  
 else:  
 self.buttons[self.save\_x][self.save\_y].config(bg='SystemButtonFace')  
 self.save\_x = x  
 self.save\_y = y  
 self.buttons[x][y].config(bg='#98FB98')  
  
 def validate\_coord(self, x, y):  
 return 0 <= x < 9 and 0 <= y < 9  
  
 def near\_color\_check(self, x, y):  
 color = self.buttons[x][y]['text']  
 for nearby\_coords in [[0, -1], [-1, -1], [-1, 0], [-1, 1]]:  
 self.line\_calculate(x, y, nearby\_coords, color)  
  
 def line\_calculate(self, x, y, nearby\_coords, color):  
 count = 1  
 x\_offsets = x  
 y\_offsets = y  
 while count < 9:  
 x\_offsets += nearby\_coords[0]  
 y\_offsets += nearby\_coords[1]  
 if self.validate\_coord(x\_offsets, y\_offsets) and color == self.buttons[x\_offsets][y\_offsets]['text']:  
 count += 1  
 else:  
 break  
 x\_offsets = x  
 y\_offsets = y  
 while count < 9:  
 x\_offsets -= nearby\_coords[0]  
 y\_offsets -= nearby\_coords[1]  
 if self.validate\_coord(x\_offsets, y\_offsets) and color == self.buttons[x\_offsets][y\_offsets]['text']:  
 count += 1  
 else:  
 break  
 if count >= 5:  
 self.score\_calculate(count)  
 x\_offsets = x  
 y\_offsets = y  
 while True:  
 x\_offsets += nearby\_coords[0]  
 y\_offsets += nearby\_coords[1]  
 if self.validate\_coord(x\_offsets, y\_offsets) and color == self.buttons[x\_offsets][y\_offsets]['text']:  
 self.delete\_ball(x\_offsets, y\_offsets)  
 else:  
 break  
 x\_offsets = x  
 y\_offsets = y  
 while True:  
 x\_offsets -= nearby\_coords[0]  
 y\_offsets -= nearby\_coords[1]  
 if self.validate\_coord(x\_offsets, y\_offsets) and color == self.buttons[x\_offsets][y\_offsets]['text']:  
  
 self.delete\_ball(x\_offsets, y\_offsets)  
 else:  
 break  
 self.delete\_ball(x, y)  
  
 def delete\_ball(self, x, y):  
 self.buttons[x][y]['text'] = ' '  
 self.buttons[x][y].config(image=self.empty)  
  
 def score\_calculate(self, count):  
 if count == 9:  
 self.score += 100  
 elif count == 8:  
 self.score += 50  
 elif count == 7:  
 self.score += 25  
 elif count == 6:  
 self.score += 10  
 elif count == 5:  
 self.score += 5  
 self.score\_label.config(text=f'Счёт:{self.score}')  
  
 def pathfinding(self, x, y):  
 goal = [x, y]  
 reachable = [[self.save\_x, self.save\_y]] *#Нужно про1ти* explored = [] *#пройдено* while len(reachable) != 0:  
 node = self.choose\_node(reachable, goal)  
 if node == goal:  
 return True  
 reachable.remove(node)  
 explored.append(node)  
 new\_reachable = []  
 for j in self.get\_adjacent\_nodes(node):  
 if j not in explored:  
 new\_reachable.append(j)  
 for adjacent in new\_reachable:  
 if adjacent not in reachable:  
 reachable.append(adjacent)  
 return False  
  
 def get\_adjacent\_nodes(self, node):  
 adjacent\_nodes = []  
 for x, y in [[1, 0], [-1, 0], [0, 1], [0, -1]]:  
 x\_offset = node[0] + x  
 y\_offset = node[1] + y  
 if (9 > x\_offset >= 0 and 9 > y\_offset >= 0) and self.buttons[x\_offset][y\_offset]['text'] == ' ':  
 adjacent\_nodes.append([x\_offset, y\_offset])  
 return adjacent\_nodes  
  
 def choose\_node(self, reachable, goal):  
 min\_cost = 20  
 best\_node = None  
 for node in reachable:  
 total\_cost = abs(goal[0] - node[0]) + abs(goal[1] - node[1])  
 if min\_cost > total\_cost:  
 min\_cost = total\_cost  
 best\_node = node  
 return best\_node  
  
 def write\_score(self):  
 scores ={}  
 with open('ScoreFile.txt', 'r') as f:  
 lines = f.readlines()  
 for line in lines:  
 if line == '':  
 break  
 readline\_score = line.split(':', 2)  
 scores[readline\_score[1][:-1]] = int(readline\_score[0])  
 if scores.get(self.login\_user, 0) > self.score:  
 messagebox.showerror('Отказ сохранения','Ваш текущий рекорд меньше, чем сохраненный ранее')  
 return  
 scores[self.login\_user] = self.score  
  
 with open('ScoreFile.txt', 'w+') as f:  
 for login in sorted(scores, key=scores.get, reverse=True):  
 f.write(str(scores[login]) + ':' + login + '\n')  
 f.close()  
 messagebox.showinfo('Сохранение','Рекорд сохранен.')  
  
Main()