**Санкт-Петербургский политехнический университет Петра Великого**

1. **Институт кибербезопасности и защиты информации**

КУРСОВАЯ РАБОТА

1. **Игра “Балда”**
2. по дисциплине «Структуры данных»
3. Выполнил
4. студент гр. 4851003/00001 Лещинский Р.Г.

<*подпись*>

1. Руководитель
2. асс. преподавателя Панков И.Д.

<*подпись*>

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2021г

1. Санкт-Петербург
2. 2021

Содержание

|  |  |
| --- | --- |
| Введение……………………………………………………………………… | 3 |
| 1. Основные файлы для работы игры…………………………………... | 4 |
| 1. Функции Wincon……………………………………………………… | 8 |
| 1. Опции меню…………………………………………………………… | 9 |
| 1. Анимация в игре………………………………………………………. | 12 |
| 1. Процесс игры………………………………………………………….. | 16 |
| 1. Реализация некоторых процессов……………………………………. | 20 |
| Заключение…………………………………………………………………… | 24 |
|  |  |
|  |  |

**Введение**

Курсовая работа подразумевала написание консольной игры на языке C – “Балда”. При написании должна была использоваться библиотека Wincon, предоставляющая функции для работы в консоли.

Игра заключается в выстраивании слов на поле 5x5, за каждое выстроенное слово начисляются баллы в зависимости от протяженности нового слова. Побеждает игрок с наибольшим количеством баллов.

Помимо самой игры, в программе курсовой работы реализованы: “справка”, “рекорды”, “О программе”. Также реализованы сложности – “2 игрока”, “против компьютера”. Добавлена авторская анимация.

**Основные файлы для работы игры**

Для работы игры используются сторонние файлы, дающие возможность сохранять информацию и использовать ее в последующих запусках. А именно:

* VOC.txt – основной словарь игры, находящийся всегда в той же папке, что и само приложение. В нем расположены ~11000 слов, каждое на отдельной строке.
* User’s\_VOC.txt – словарь пользователя, где он может добавить слова для игры. Расположение слов в нем такое же, как и в основном словаре. Файл пользовательского словаря можно поменять – “Справка” – “Словарь пользователя”. В случае ввода несуществующего текстового файла, старый файл пользовательского словаря не изменится.

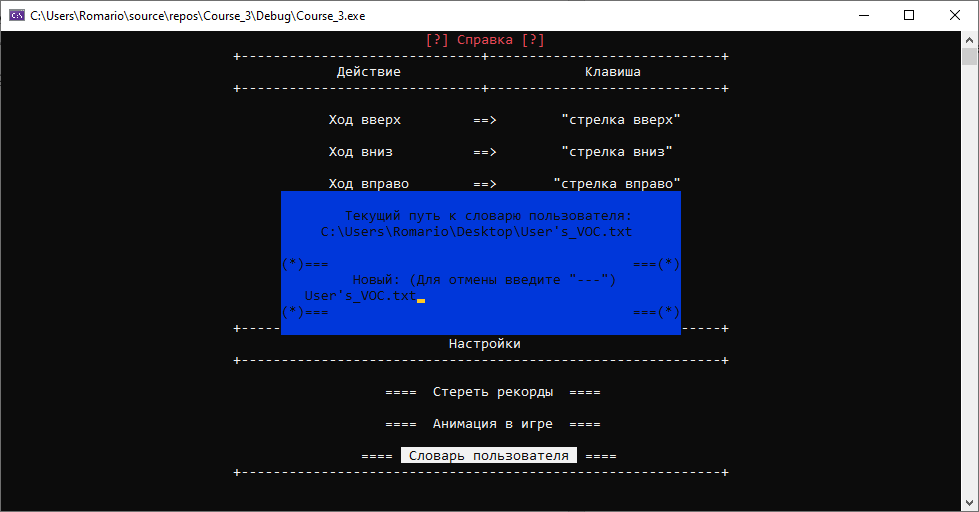


Рисунок 1 – Интерфейс изменения пути к пользовательскому словарю

* LAST\_GAME.txt – файл предыдущей игры. Другими словами, в нем хранятся данные прошлой игры: количество очков, игровое поле со выставленными буквами, слова в блокноте, количество пропущенных ходов, имя игрока, уровень сложности, используемые слова. Для координации по файлу, заранее известны расположения данных, а также введены брейкпоинты “|” для упрощения.

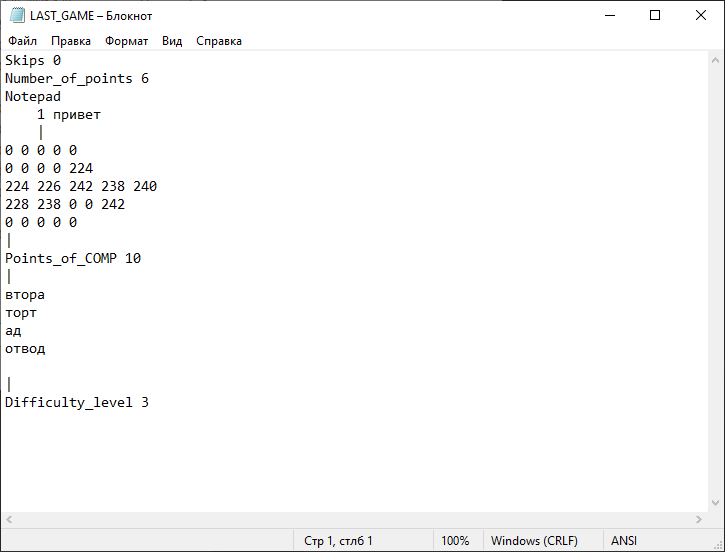


Рисунок 2 – Расположение данных в файле LAST\_GAME

* RECORDS.txt – файл с рекордами предыдущих игр.
* Data.txt – файл с основными данными всей программы. В нем располагаются значение Random переменной, значение Animation, User\_dictionary, Last\_Game.

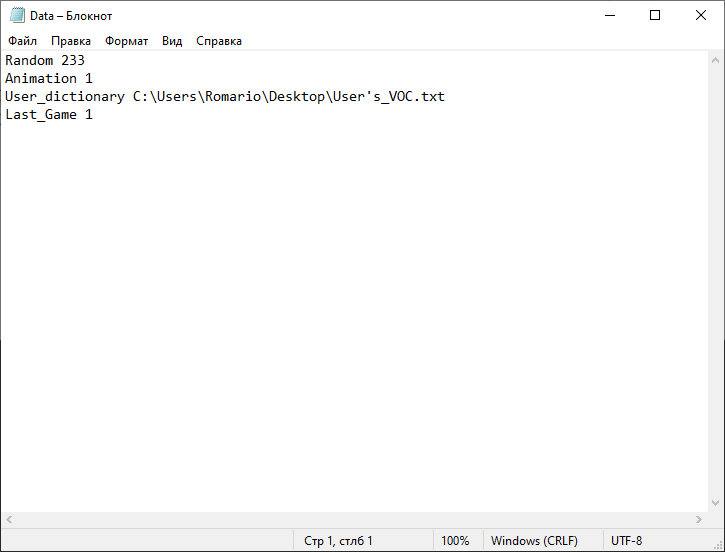


Рисунок 3 – Расположение данных в файле Data.txt

Random переменная хранит в себе динамическое значение, всячески меняющееся во время работы программы. Меняется значение посредством использования написанной функции LKG() (Линейный конгруэнтный генератор). За счет этого значения показывается определенная анимация в процессе игры, выбирается подсказка на загрузочном экране, компьютер выбирает слово.

Animation переменная показывает наличие включенных показываемой анимации. Поскольку загрузочные экраны также являются анимацией, то добавлена возможность отключать их показ. (“Справка” – “Анимация в игре”) (Об анимации ниже).

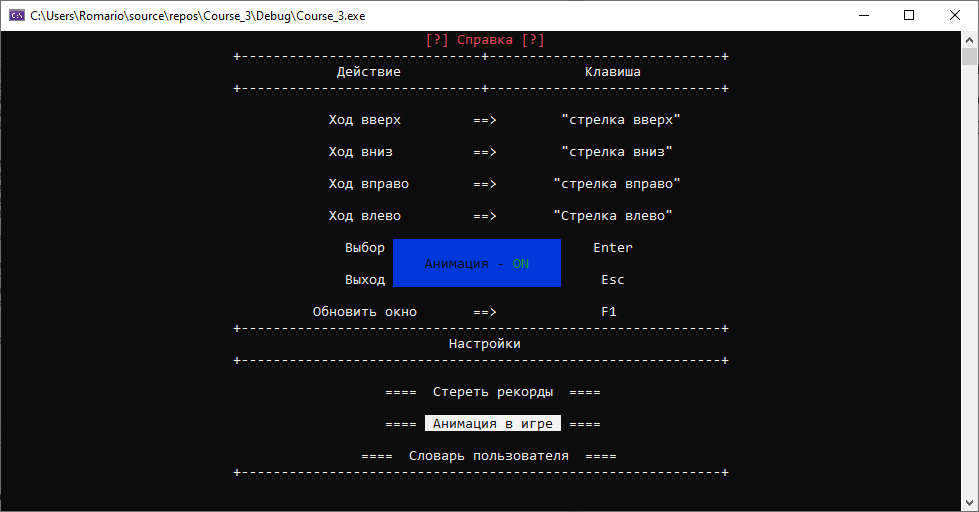


Рисунок 4 – Интерфейс меню включения/отключения анимации в игре

User\_dictionary – путь к пользовательскому словарю. Изначальный путь – User’s\_VOC.txt.

Last\_Game – наличие незавершённой игры. В случае значения 1 будет считываться файл LAST\_GAME.txt.

Во всех файлах соблюдается строгий порядок расположения данных. В случаях корректировки файлов LAST\_GAME, Data.txt – во время считывания данных и проверки их расположения в файле будет выведен экран, с предупреждением о повреждении файла. После игра закрывается. В случае ошибок, следует удалить поврежденные файлы, и снова запустить игру. Поскольку в случае отсутствия файлов игры создаются стандартные их версии, то проблема будет решена, однако, очевидно, будет потеря тех или иных данных настроек.

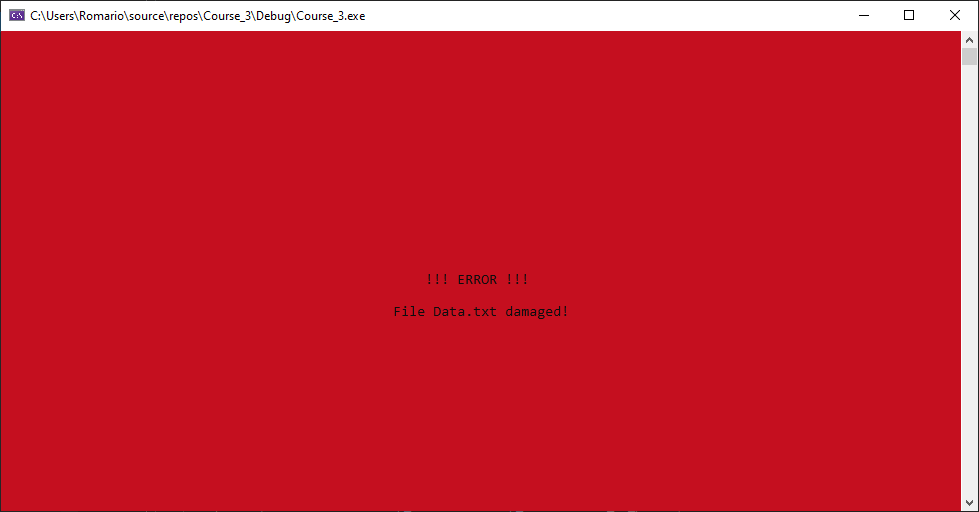


Рисунок 5 – Ошибка считывания файла

Для хранения данных из файла Data.txt введена дополнительная структура, в которой записаны все значения переменных и весь текст из файла в массиве типа char.

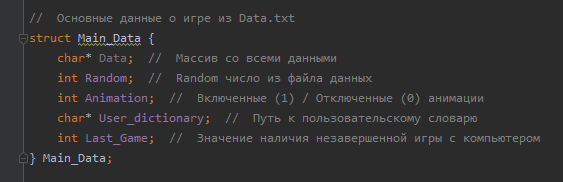


Рисунок 6 – Структура данных Data.txt

**Функции Wincon**

Wincon – выданная преподавателем библиотека с набором функций для реализации игры.

con\_init() – функция инициализации консоли и ее настройки. Используется в начале работы программы.

con\_width\_new() и con\_height\_new() - новый добавленные функции в библиотеку на замену функциям con\_width() и con\_height() – функций получения текущего размера консоли (ширина и высота соответственно). Используются функции почти везде, поскольку для создания анимации требуется выводить символы картинки лишь в определенных местах, также для отрисовки интерфейсов ровно по центру.

symbol\_code() – также добавленная функция в библиотеку. Функция добавляет новую возможность ввода символов. Используется при вводе имени в конце игры.

con\_set\_color() – функция изменения цвета шрифта и фона этого шрифта. Также используется почти в каждой части программы для отрисовки анимации, меню, интерфейсов и т.п.

clrscr() – Функция очищения всей консоли. В совокупности с установленным цветом фона (через con\_set\_color() ) может окрасить всю консоль в новый цвет.

gotoxy() – передвижение курсора в определённую точку на экране консоли. Поскольку интерфейсы, картинки анимации – массивы строк, то отрисовка по центру осуществляется перебросом курсора в определенную часть консоли и выводом очередной строки.

show\_cursor() – функция отображения курсора. Используется в случаях, когда не требуется курсор.

key\_is\_pressed() и key\_pressed\_code() – функции для работы над нажатыми клавишами. Первая функция срабатывает, когда нажат символ, вторая – возвращает код нажатого символа. В основном, применяются эти функции в связке с циклом while() для работы в меню, а также для использования не буквенных клавиш.

**Опции меню**

В меню находится 5 опций – “Играть”, “Рекорды”, “О программе”, “Справка” и “Выход”.

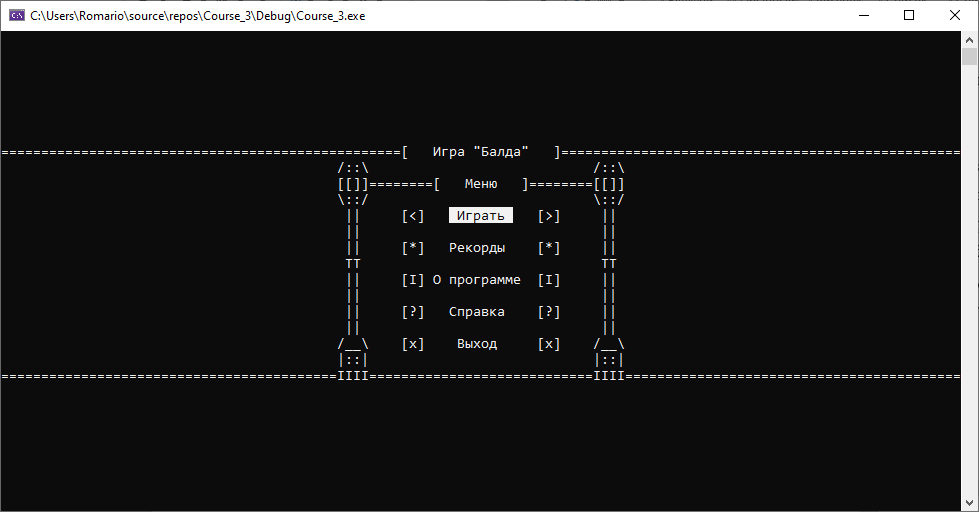


Рисунок 7 – Опции меню игры

- Опция “Играть” выводит меню выбора игры – “Против компьютера” и “2 игрока”. Опция “Продолжить последнюю” появляется лишь при наличии незаконченной игры. Подробнее о самой игре ниже.

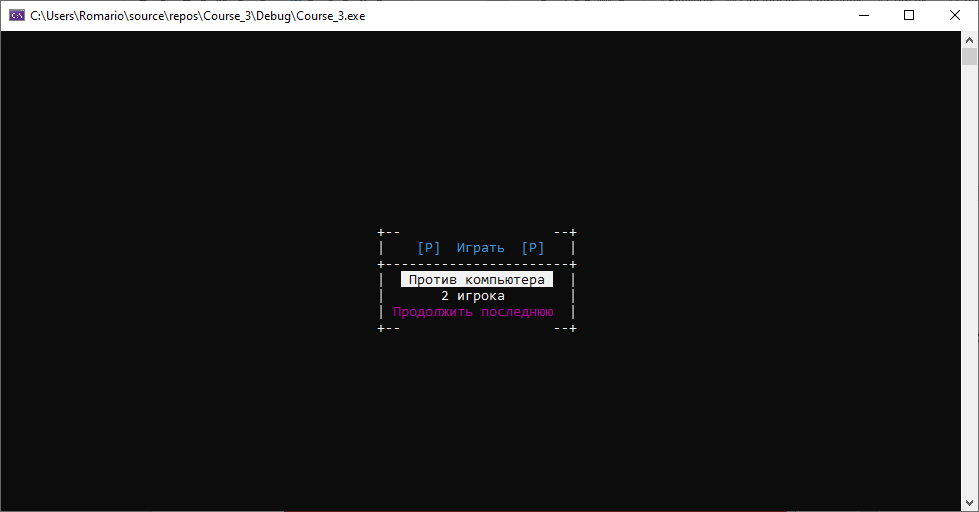


Рисунок 8 – Опция меню “Играть”

- Опция “Рекорды” после экрана анимации показывает записанные рекорды и имена игроков.

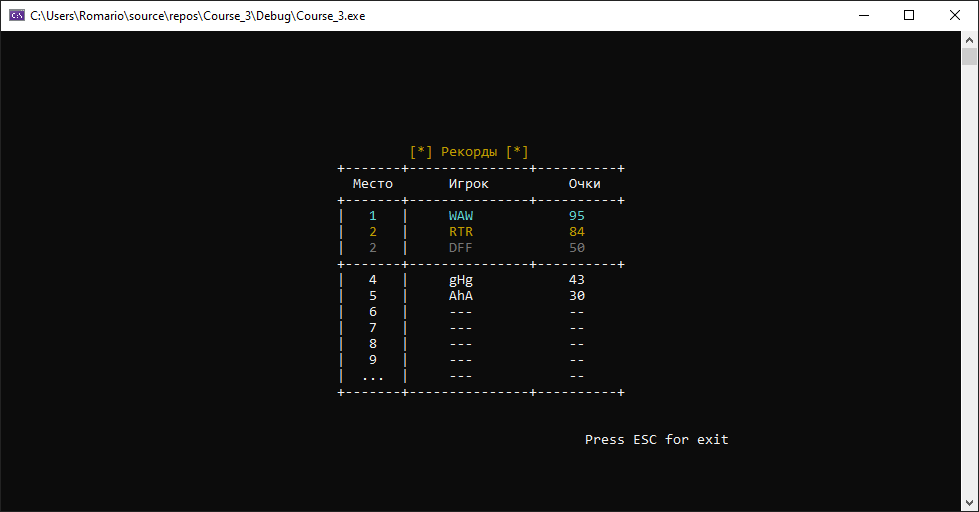


Рисунок 9 – Таблица рекордов

В данной таблице представлены рекорды и из игры 2-ух игроков и из игры против компьютера.

- Опция “О программе” предоставляет информацию о создателях и год создания

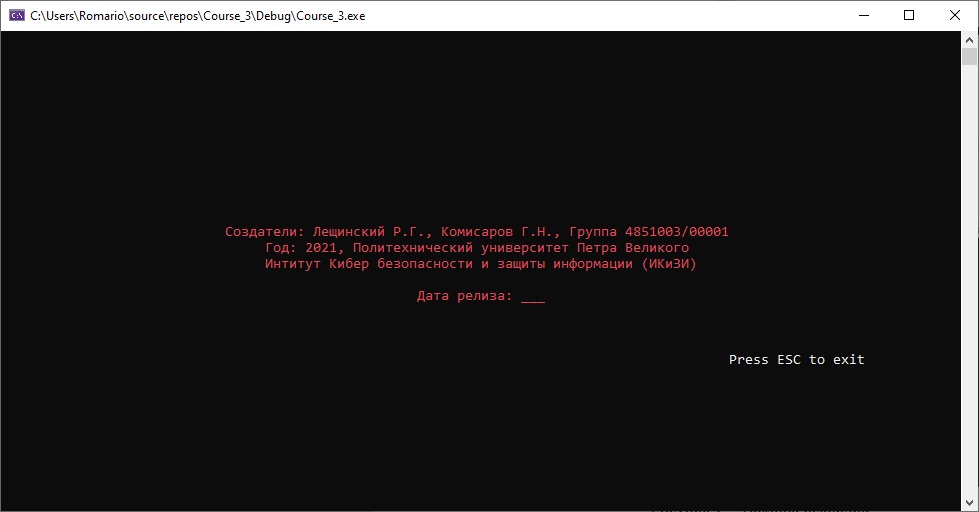


Рисунок 10 – Опция “О программе”

- Опция “Справка” показывает используемые клавиши в игре, а также настройки. В настройках можно стереть рекорды, отключить анимацию в игре и поменять путь к словарю пользователя.

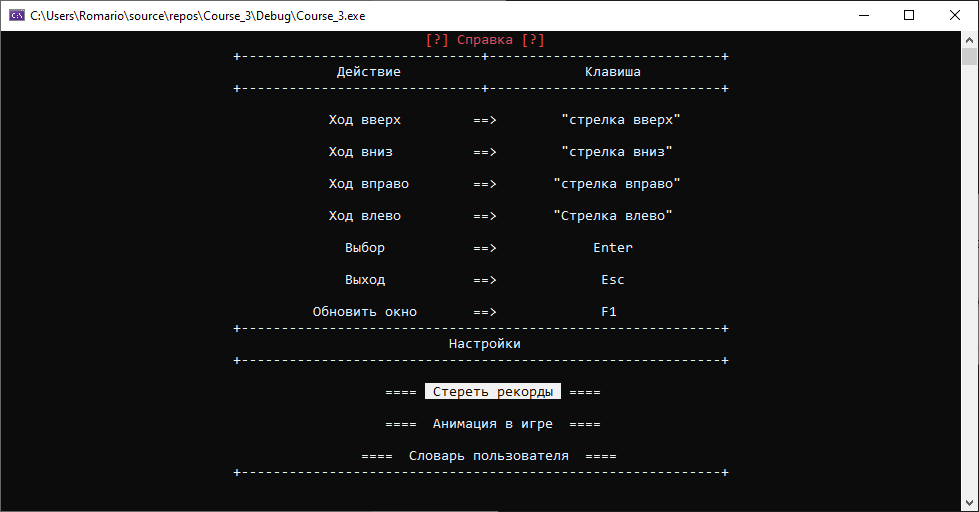


Рисунок 11 – Опция “Справка”

**Анимация в игре**

В игре также присутствует 6 анимаций, каждая из которых описывается в отдельных функциях.

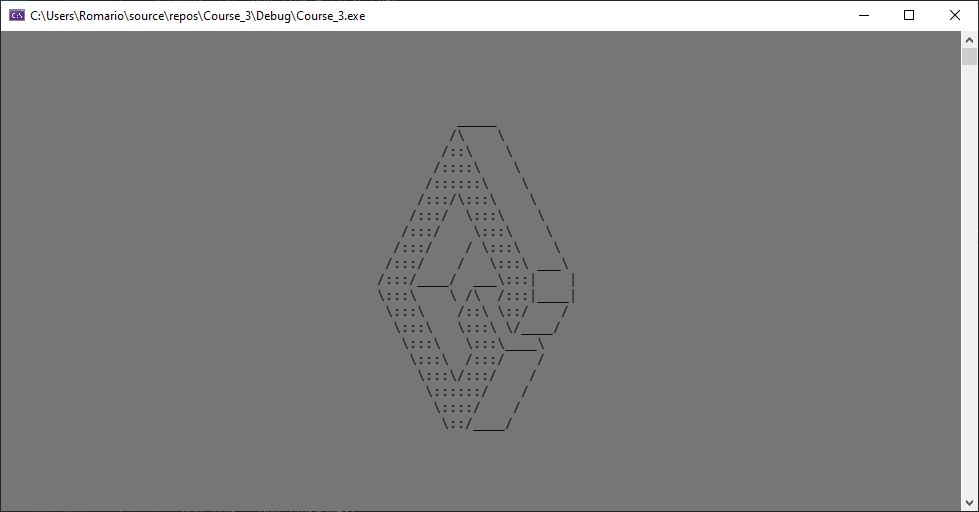


Рисунок 12 – Авторская анимация (1)

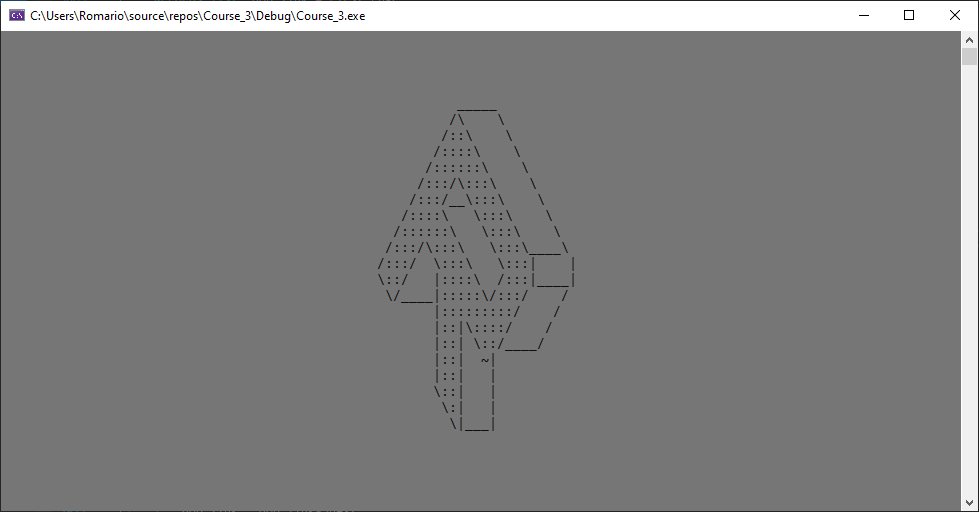


Рисунок 13 – Авторская анимация (2)

- Авторская анимация воспроизводится при запуске игры. Из себя представляет переход буквы “G” в “R” или наоборот в зависимости от значения переменной Random.

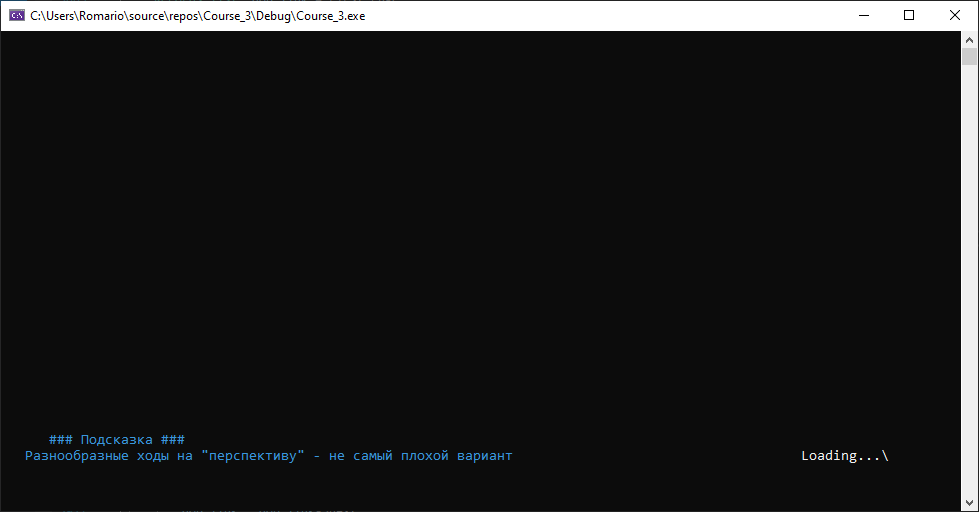


Рисунок 14 – Основная анимация загрузки

- Основная анимация загрузки представляет из себя анимацию в нижнем правом углу и определенную подсказку в нижнем левом (в зависимости от значения Random. Всего подсказок - 8). Из себя такой “Загрузочный экран” представляет лишь анимацию, он не отображает процесс загрузки чего-либо. Присутствует в начале работы программы и при запуске игры.



Рисунок 15 – Анимация загрузки рекордов

- Анимация загрузки рекордов аналогична основной анимации – Кроме как анимации никаких функций в себе не несет. Запускается при загрузке рекордов.



Рисунок 16 – Анимация окончания игры

- Анимация окончания игры показывает окончательные результаты игры и победителя. После просит ввести имя победителя для записи в таблицу рекордов.

Оставшиеся 2 анимации являются анимацией появления и закрытия главного меню.

Работа анимации достаточно проста – она разбивается на некоторое количество картинок, постепенно сменяющих друг друга. На рисунке 17 представлен пример работы анимации на примере авторской. Вырисовка оставшихся происходит по схожему алгоритму.

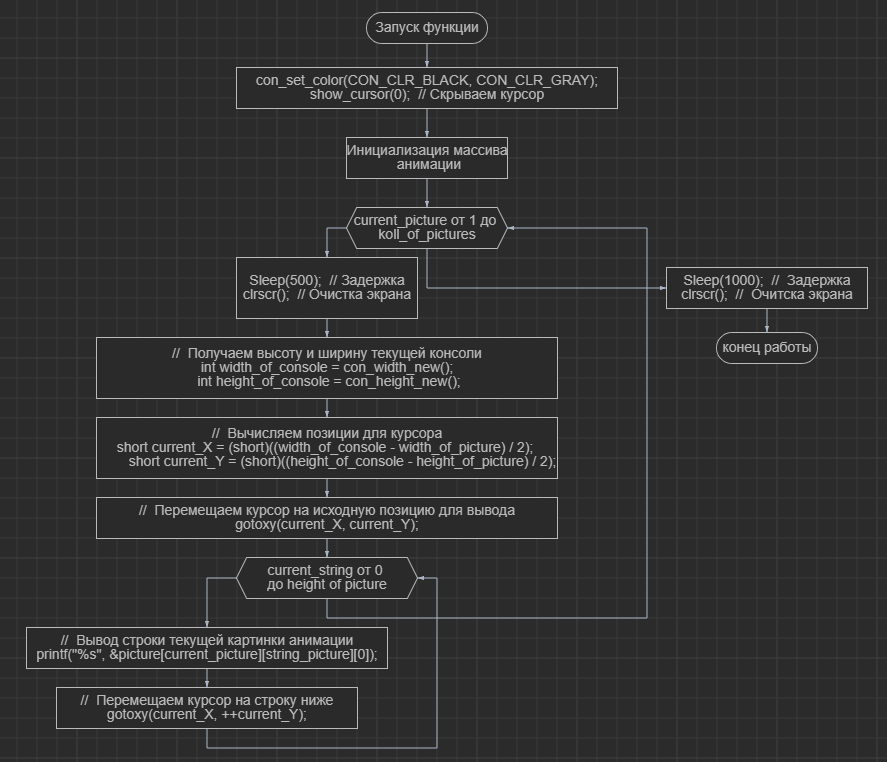


Рисунок 17 – Блок-схема рисовки анимации на примере авторской анимации (в начале работы игры)

**Процесс игры**

В программе реализовано 2 режима игры: 2 игрока и против компьютера. У режима игры против компьютера также есть сложности – Высокая, средняя и низкая. О работе логики компьютера позже (в разделе **реализации некоторых процессов**).



Рисунок 18 – Сложности игры против компьютера

После выбора сложности программа просит ввести начальное слово, которое будет располагаться посередине игрового поля. После ввода слова выводится основное поле.

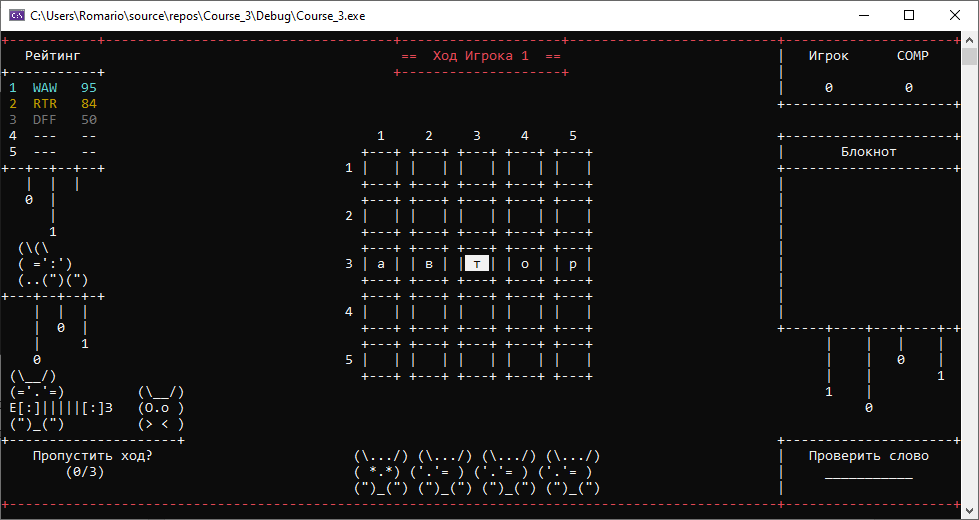


Рисунок 19 – Игровое поле

На нем присутствуют: Счет игроков (верхний правый угол), блокнот для записи потенциальных слов (центр справа), проверка слов (нижний правый угол), возможность пропуска хода (нижний левый угол) и записанные рекорды в верхнем левом углу. Для передвижения используются клавиши направления (стрелочки). Для ввода слова, по правилам игры, нудно добавить на поле 1 букву и использовать ее в вводимом слове. Для выбора используется клавиша Enter. При выставлении новой буквы, ее ячейка окрашивается в синий цвет.

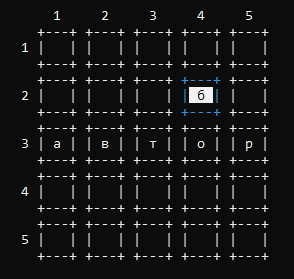


Рисунок 20 – Выставленная буква

После выставления буквы следует выбора слова. Слово набирается последовательным выбором каждой буквы (выбираются буквы клавишей Enter)

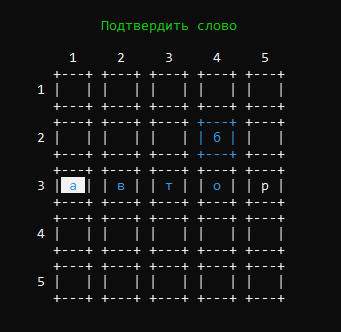


Рисунок 21 – Набор слова

После выделения выставленной буквы появляется функция “Подтвердить слово”. Для снятия выделения выбранных букв и сброса выбранного слова используется клавиша Esc. После подтверждения слова, ход передается противнику.

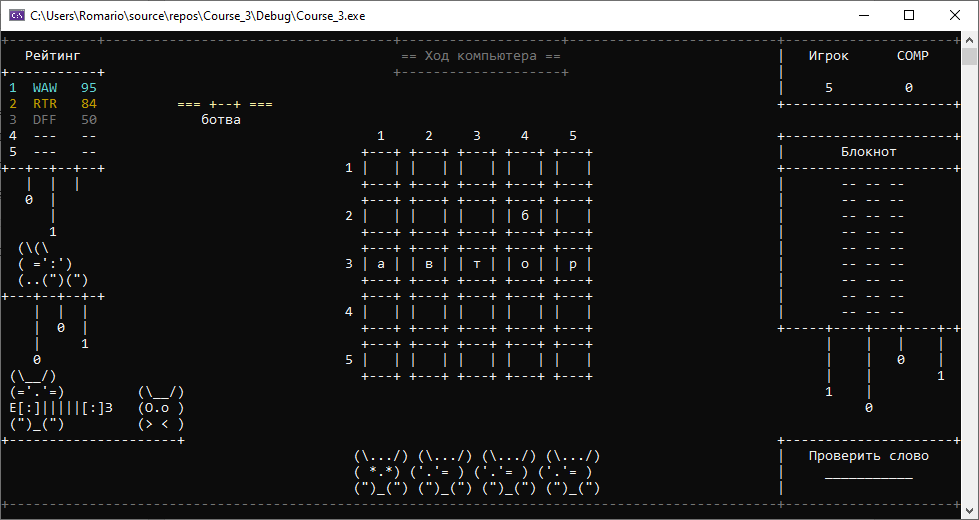


Рисунок 22 – Ход компьютера

После передачи хода, части интерфейса перекрашиваются в цвет противника (Против компьютера – серый, против игрока - синий). После передачи очереди компьютеру, игрок будет наблюдать статичный экран 3 секунды, якобы требуемые для хода компьютера. Однако в программе специально поставлена задержка в 3 секунды для большей плавности игры. Иначе компьютер мгновенно выбирает и выставляет слово на поле. Также, как видно на рисунке 22, появляется история введенных слов. Новые введенные слова всякий раз буду появляться в начале списка. Предыдущие слова же будут спускаться ниже. Чтобы слова не налезали на интерфейс, в истории видны лишь последние 10 слов.

В случае выбора уже использованного слова, будет выведено сообщение о невозможности его использовании.

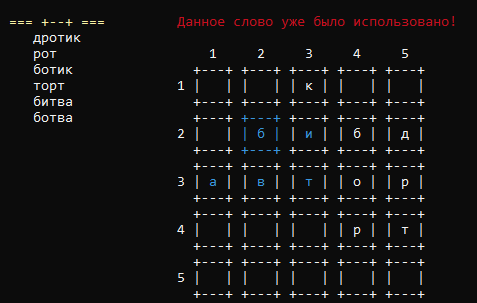


Рисунок 23 – Сообщение о невозможности ввести уже использованное слово ранее

Как только все поле будет заполнено символами, игра заканчивается. Выводятся окончательные очки игроков и вводится имя победителя (кроме случая, когда побеждает компьютер).

**Реализации некоторых процессов**

- Выбор слов компьютером происходит в несколько этапов: составления списка возможных для использования слов, проверка их на используемость ранее, в зависимости выбор слова по его длине. Все этапы происходят в 4-ух функциях: COMP\_COMPStep(), Game\_Words(), Game\_WordInField() и Game\_CheckAddingWord(). Функция Game\_Words() подбирает слова и возвращает двумерный массив-список выбранных слов. Game\_WordInField() – рекурсивная функция, пытающаяся расположить слово всеми возможными способами на поле. COMP\_COMPStep() уже из списка выбранных слов выбирает окончательное. В зависимости от сложности, расставляются приоритеты выбора слов (процентами). Game\_CheckAddingWord() – функция проверки подстановки слова на игровом поле.

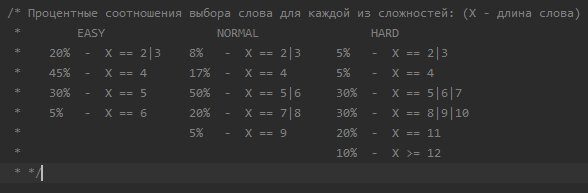


Рисунок 24 – Проценты для выбора слова определенной длины в зависимости от сложности

Относительно значения Random и используемых процентов, подбирается слово. Значение Random также меняет свое значение всякий раз, как будет выбрано слово компьютером. Такой способ достаточно простой, однако требует многих проверок на правильность соотношений, а также очень сильно зависит от реализованного генератора псевдослучайных чисел. На рисунках 25.1, 25.2, 25.3 и 25.4 представлены алгоритмы подбора слов компьютером.

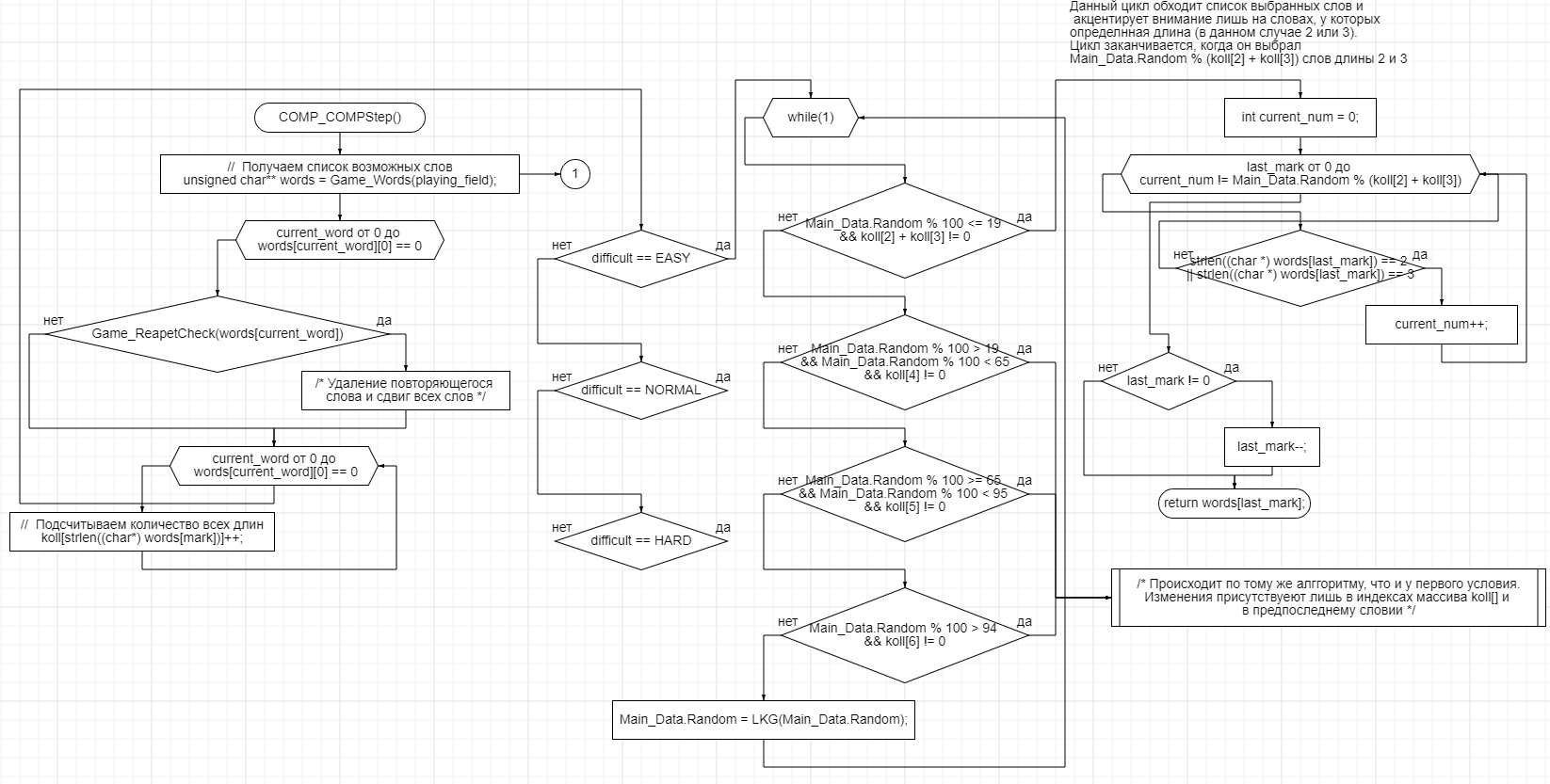


Рисунок 25.1 – Блок-схема выбора слова компьютером (функция COMP\_COMPStep())

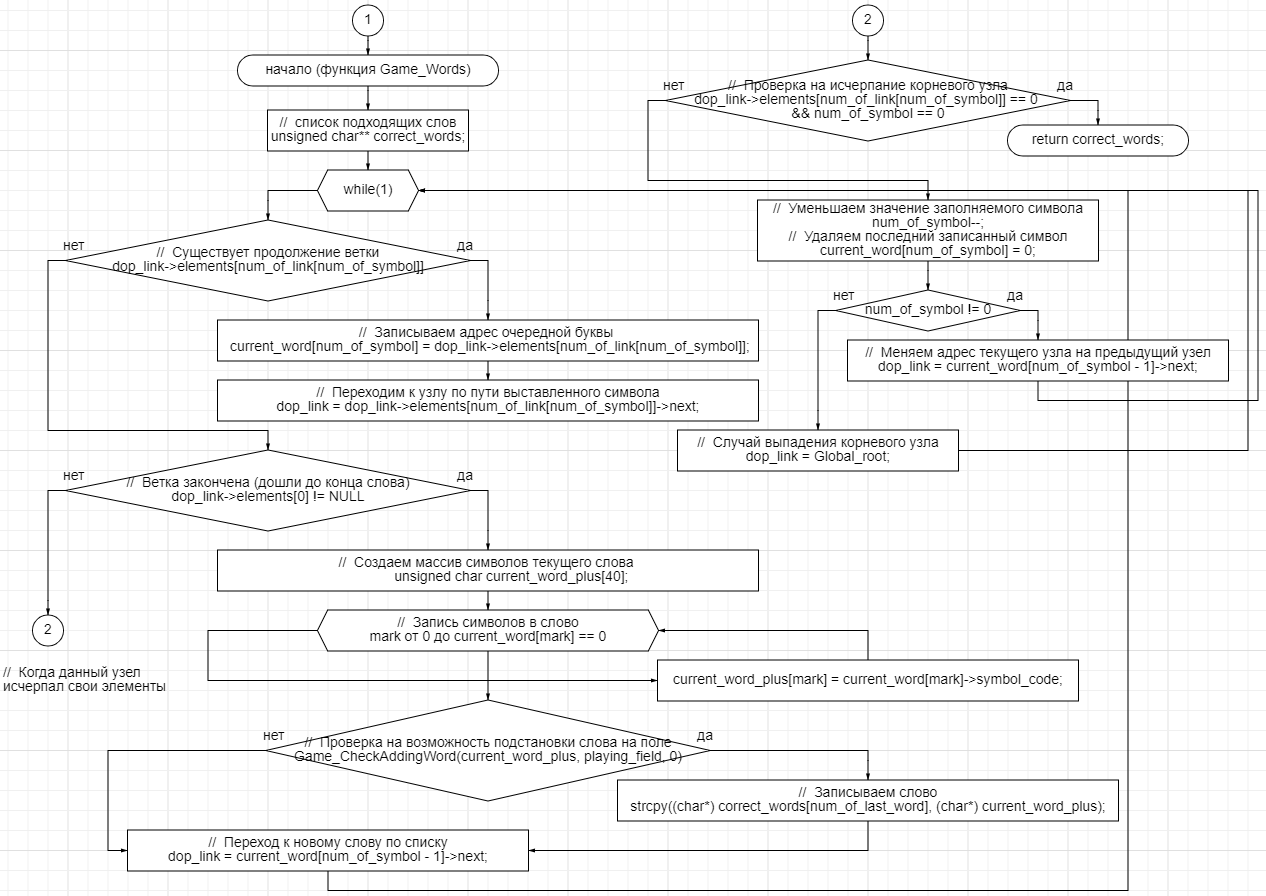


Рисунок 25.2 – Блок-схема выделения слов из префиксного дерева (функция GAME\_Words())

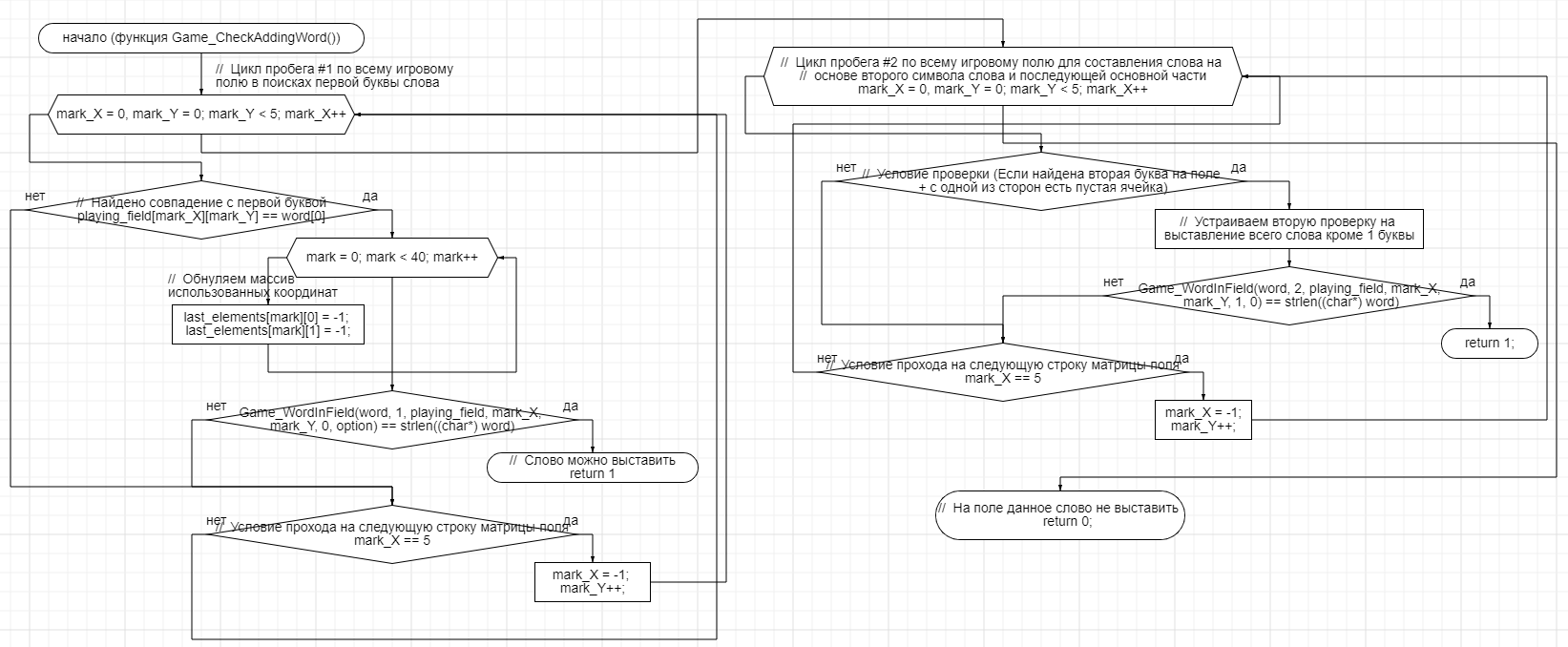


Рисунок 25.3 – Блок-схема функции проверки подстановки слова на игровом поле (функция Game\_CheckAddingWord())

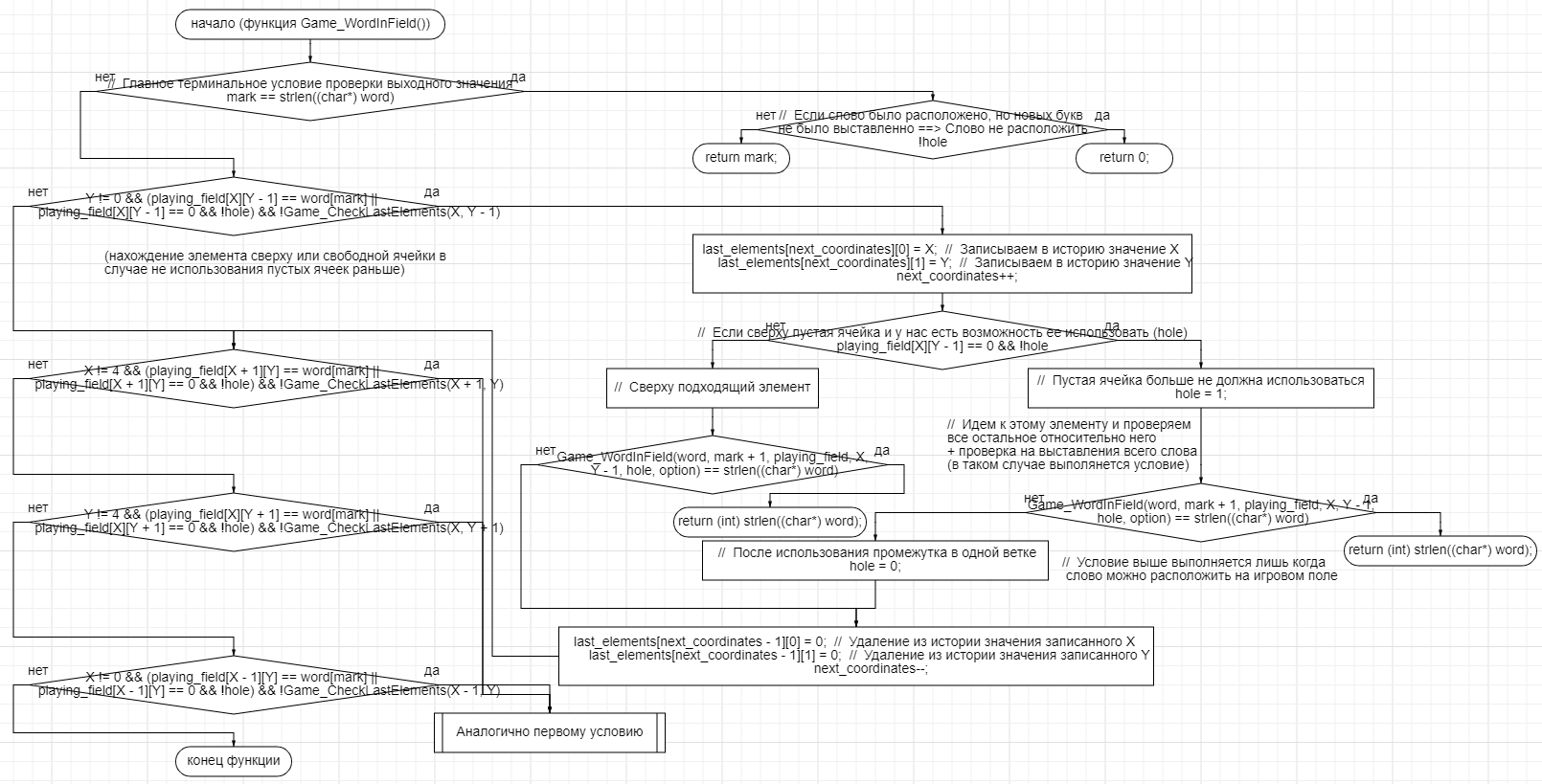


Рисунок 25.4 – Блок-схема рекурсивной функции поиска букв слова

- Хранение общего словаря (главный словарь + пользовательский) осуществляется в префиксном дереве (trie). На рисунке 25 представлена его образная схема работы. Каждый узел trie дерева является списком букв, входящим в него. Каждая буква при этом должна ссылаться на другой узел дерева с последующими буквами. То есть узел дерева – структура со списком элементов-букв, которые также являются структурами и включают в себя целочисленное значение буквы и адрес на последующий адрес.

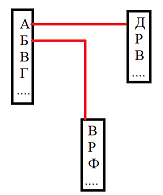


Рисунок 26 – Образная схема trie дерева

Для поиска и вставки слова слова достаточно выбрать в каждом узле следущую букву из слова. То есть, при поиске слова “выбор”, то в первом узле мы находим букву “в” и переходим по ее адресу к следующему узлу. В слудеющем узле находим букву “ы” и так далее до нахождения в последней букве значение адреса нулевого элемента адресов равнялся NULL. Это сделано для того, чтобы можно было различать 2 слова, один и которых является частью предыдущего (слова “вор” и “воск”). В других ситуациях используются заглушки-узлы. Исходя из надобности находить “исключительные слова”, в массиве адресов нулевой элемент отводится для понимания является ли данная буква концом слова или нет.

- Хранение списка использованных слов и рекордов игроков происходит в одностороннем линейном списке.

- Под хранение значений режима игры (“TWO\_PLAYERS” – два игрока, “EASY” – легкая сложность с компьюетром, “NORMAL” – средняя сложность…), хранение данных об игрока в текущей сессии игры, текущая подсвечиваемая опция на поле игры и т.п. выделены структуры (записи).

**Заключение**

Реализация игры была достаточно простой в плане сложности написания, поскольку все функции из библиотеки Wincon были достаточно понятны. Однако некоторые функции в библиотеке не работали корректно и поэтому их пришлось заменить. Также в библиотеку были добавлены новые для работы программы.

Реализованный ЛКГ имеет неоптимальные значения коэффициентов, что порождает не совсем правильное поведение при выборе слов компьютером.

Функции записи в файл данных могут быть улучшены, добавлены проверки написания словарей. Также возможно улучшение реализации trie дерева, а именно удаление узлов-заглушек, поскольку данных в одном узле дерева достаточно для определения/нахождения слова.