

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования «Самарский национальный исследовательский
университет имени академика С.П. Королева (Самарский университет)»

Институт _____ информатики и кибернетики _____

Кафедра _____ программных систем _____

ОТЧЕТ

_____ к лабораторному практикуму _____
_____ по дисциплине «Технологии программирования» _____
_____ по теме «Автоматизированная система составления и _____
_____ разгадывания классического кроссворда по выбранной теме» _____

Обучающийся _____ А.А. Малышев

Обучающийся _____ А.А. Манакова

Обучающийся _____ А.М. Федоров

Обучающийся _____ В.И. Хорина

Руководитель _____ Л.С. Зеленко

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение
высшего образования «Самарский национальный исследовательский
университет имени академика С.П. Королева (Самарский университет)»

Институт _____ информатики и кибернетики _____

Кафедра _____ программных систем _____

ЗАДАНИЕ

на лабораторный практикум по дисциплине
«Технологии программирования»
обучающимся в группе № 6403-090301D

А.А. Малышеву

А.А. Манаковой

А.М. Федорову

В.И. Хориной

- 1 Тема проекта: «Автоматизированная система составления и разгадывания классического кроссворда по выбранной теме»
- 2 Исходные данные к проекту: см. приложение к заданию
- 3 Перечень вопросов, подлежащих разработке:
 - 3.1 Произвести анализ предметной области: изучить основные принципы составления кроссвордов, изучить алгоритмы генерации кроссвордов
 - 3.2 Выполнить обзор существующих систем-аналогов
 - 3.3 Разработать информационно-логический проект системы
 - 3.4 Разработать и реализовать программное и информационное обеспечение, провести его тестирование и отладку.
 - 3.5 Оформить документацию проекта
 - 3.6 Подготовить презентацию по разработанной системе
- 4 Перечень графических разработок
 - 4.1 Структурная схема системы
 - 4.2 Диаграмма классов, диаграмма модулей (компонентов)
 - 4.3 Схемы основных алгоритмов

5 Календарный план выполнения работ

№ п/ п	Содержание работы по этапам	Объем этапа в % к общему объему проекта	Срок окончания	Фактическое выполнение
1	Оформление технического задания и его утверждение	5	17.09.2021	
2	Описание и анализ предметной области (1 раздел)	10	01.10.2021	
3	Проектирование системы (2 раздел)	30	26.11.2021	
3.1	Разработка структурной схемы системы	5	15.10.2021	
3.2	Разработка функциональной спецификации системы	10	29.10.2021	
3.3	Разработка прототипов экранных форм	10	29.10.2021	
3.4	Разработка основных алгоритмов	5	26.11.2021	
4	Реализация проекта, разработка контрольных примеров. Предъявление реализации руководителю (3 раздел)	45	10.12.2021	
5	Корректировка проекта и оформление документации проекта. Защита проекта с представлением презентации	10	24.12.2021	

Задание принял

к исполнению _____ 03.09.2021 А.А. Малышев

_____ 03.09.2021 А.А. Манакова

_____ 03.09.2021 А.М. Федоров

_____ 03.09.2021 В.И. Хорина

ПРИЛОЖЕНИЕ

к заданию на лабораторный практикум
обучающимся в группе № 6403-090301D

А.А. Малышеву

А.А. Манаковой

А.М. Федору

В.И. Хориной

Тема проекта: «Автоматизированная система составления и разгадывания
классического кроссворда по выбранной теме»

Исходные данные к проекту:

1 Характеристика объекта автоматизации:

- 1) объект автоматизации: классический кроссворд;
- 2) виды автоматизируемой деятельности:
 - процесс авторизации/регистрации пользователей;
 - процесс составления кроссворда;
 - процесс генерирования кроссворда;
 - процесс разгадывания кроссворда;
 - процесс работы со словарем понятий;
 - процесс визуализации работы с кроссвордом;
- 3) количество ролей пользователей – 2;
- 4) минимальная длина пароля – 4 символа;
- 5) максимальная длина пароля – 12 символов;
- 6) минимальная длина логина – 2 символа;
- 7) максимальная длина логина – 8 символов;
- 8) минимальный размер сетки по горизонтали (клеток) – 15;
- 9) максимальный размер сетки по горизонтали (клеток) – 30;
- 10) минимальный размер сетки по вертикали (клеток) – 10;
- 11) максимальный размер сетки по вертикали (клеток) – 25;
- 12) минимальная длина одного слова – 3 символа;
- 13) максимальная длина одного слова – 15 символов;

- 14) количество видов сортировки словаря понятий – 2;
 - 15) количество способов создания кроссворда – 2;
 - 16) количество языков записи понятий – 1.
- 2 Требования к информационному обеспечению:
- 1) информационное обеспечение разрабатывается на основе следующего источника:
 - Описание структуры кроссворда [Электронный ресурс]. URL: <https://ru.wikipedia.org/wiki/Кроссворд> (дата обращения: 11.09.2021);
 - 2) структура словаря понятий (понятие и его определение располагаются в одной строке, разделены пробелом);
 - 3) словари понятий хранятся в текстовых файлах формата *.dict;
 - 4) база данных разрабатывается на основании следующих сведений:
 - о пользователях (логин, пароль);
 - 5) кроссворды хранятся в файлах, структура файла определяется в процессе проектирования.
- 3 Требования к техническому обеспечению:
- 1) тип ЭВМ – IBM PC совместимый;
 - 2) монитор с разрешающей способностью не ниже 800 x 600;
 - 3) манипулятор – мышь;
 - 4) технические характеристики определяются в процессе выполнения проекта.
- 4 Требования к программному обеспечению:
- 1) тип операционной системы – Windows 7 и выше;
 - 2) язык программирования – C#;
 - 3) среда программирования – Visual Studio 2019.
- 5 Общие требования к проектируемой системе:
- 5.1 Функции, реализуемые системой:
- 1) общесистемные функции:
 - авторизация пользователя в системе (ввод логина, пароля);

- регистрация пользователя в системе (ввод логина, пароля);
- аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
- визуализация процессов работы с кроссвордом;
- автоматическое составление кроссворда по заданным параметрам;
- проверка языка записи понятий;
- сортировка словаря по выбранному критерию;
- ручное составление кроссворда по заданным параметрам;
- фильтрация словаря понятий по маске;
- проверка правильности разгадывания кроссворда;
- выдача справочной информации о системе;

2) функции администратора:

- настройка параметров кроссворда при создании:
 - 1 задание размерности сетки;
 - 2 задание названия кроссворда;
 - 3 подключение словаря понятий;
 - 4 выбор режима составления кроссворда;
- составление/редактирование кроссворда:
 - 1 добавление слова;
 - 2 удаление слова;
 - 3 задание маски для поиска;
 - 4 изменение ориентации слова;
- сортировка словаря по алфавиту;
- сохранение кроссворда в файл заданной структуры;
- загрузка кроссворда из файла;
- работа со словарями понятий:
 - 1 добавление понятия;
 - 2 удаление понятия;

- 3 изменение понятия;
 - 4 выбор критерия сортировки;
 - 5 поиск по заданной маске;
 - 6 загрузка словаря из файла;
 - 7 сохранение словаря из файла;
 - 8 создание нового словаря понятий;
- 3) функции пользователя (игрока):
- загрузка кроссворда из файла;
 - разгадывание кроссворда:
 - 1 выбор слова;
 - 2 вписывание/удаление/изменение буквы;
 - сохранение кроссворда в файл.

5.2 Технические требования к системе:

- 1) режим работы – диалоговый;
- 2) время автоматической генерации кроссворда – не более 30 с;
- 3) система должна удовлетворять санитарным правилам и нормам СанПин 2.2.2./2.4.2198-07;
- 4) условия работы средств вычислительной техники (содержание вредных веществ, пыли и подвижность воздуха) должны соответствовать ГОСТ 12.1.005, 12.01.007;
- 5) температура окружающего воздуха – 15-35°C;
- 6) влажность воздуха – 45-75%.

Руководитель

проекта _____ 17.09.2021 _____ Л.С. Зеленко

Задание принял

к исполнению	_____	03.09.2021	_____	А.А. Малышев
	_____	03.09.2021	_____	А.А. Манакова
	_____	03.09.2021	_____	А.М. Федоров
	_____	03.09.2021	_____	В.И. Хорина

РЕФЕРАТ

Пояснительная записка 91 с, 49 рисунков, 9 таблиц, 18 источников, 2 приложения.

Графическая часть: 19 слайдов презентации PowerPoint.

КРОССВОРД, ГЕНЕРАТОР КРОССВОРДОВ, РУЧНОЙ РЕЖИМ, СЛОВАРЬ ТЕРМИНОВ, ВАРИАНТ ОТОБРАЖЕНИЯ, ПАРАМЕТРЫ КРОССВОРДА, РАЗГАДЫВАНИЕ.

Объектом автоматизации является классический кроссворд.

Во время лабораторного практикума разработаны алгоритмы и соответствующая им программа, позволяющая выполнять автоматическую генерацию классического кроссворда по заданной теме. Задания (понятие и его расшифровка) хранятся в текстовом файле и могут дополняться вручную (с использованием текстового редактора) или внутри программы, при этом ограничений на длину словаря не существует. Тема кроссворда выбирается пользователем в соответствии с содержанием словаря заданий. Программа позволяет сформировать кроссворд, учитывая ограничения на параметры. В системе имеется возможность сохранения кроссвордов в файл с целью последующего их разгадывания.

Программа написана на языке C# в среде Visual Studio 2019 и функционирует под управлением операционной системы Windows 7/8/10.

СОДЕРЖАНИЕ

Введение.....	11
1 Описание и анализ предметной области.....	14
1.1 Описание предметной области	14
1.1.1 Правила классического кроссворда	14
1.1.2 Классификация кроссвордов.....	16
1.2 Описание систем-аналогов.....	18
1.2.1 Система-аналог «Crosswordus».....	18
1.2.2 Система-аналог «Decalion»	19
1.3 Диаграмма объектов предметной области	21
1.4 Постановка задачи	23
2 Проектирование системы	26
2.1 Структурная схема системы	26
2.2 Спецификация системы.....	28
2.2.1 Функциональная спецификация	29
2.2.2 Спецификация качества.....	30
2.2.3 Перечень исключительных ситуаций	30
2.3 Разработка прототипа интерфейса пользователя системы	43
2.3.1 Режим игрока.....	45
2.3.2 Режим администрирования	46
2.4 Разработка структур данных и классов	49
2.5 Логическая модель данных	51
2.6 Разработка и описание алгоритмов обработки данных	52
2.7 Выбор и обоснование комплекса программных средств.....	55
2.7.1 Выбор языка программирования.....	55
2.7.2 Выбор операционной системы	56
2.7.3 Выбор среды программирования	57
3 Реализация системы	58
3.1 Разработка и описание интерфейса пользователя	58

3.1.1	Режим игрока.....	58
3.1.2	Режим администратора.....	60
3.2	Реализация классов и структур данных.....	65
3.3	Реализация и описание модулей программы	65
3.4	Выбор и обоснование комплекса технических средств.....	68
3.4.1	Расчет объема занимаемой памяти.....	68
3.4.2	Минимальные требования, предъявляемые к системе.....	70
	Список использованных источников	72
	Приложение А Руководство пользователя	74
A.1	Назначение системы	74
A.2	Условия работы системы.....	74
A.3	Установка системы.....	74
A.4	Работа с системой.....	74
A.4.1	Работа с системой в режиме администратора	75
A.4.2	Работа с системой в режиме игрока	80
	Приложение Б Листинг модулей программы.....	83

ВВЕДЕНИЕ

Исследователям встречались находки, похожие на кроссворд, датированные ещё I–IV вв. н. э. В частности, во время раскопок, производимых в Помпеях, была обнаружена головоломка, удивительно напоминающая современный кроссворд, которую ученые датировали 79 годом н. э. При этом существуют различные версии изобретения кроссвордов. Среди стран, претендующих на звание родины кроссвордов, Италия, Великобритания, США [1].

По одной из версий, прототипы современных кроссвордов появились ещё в XIX веке. Самый первый дошедший до нас кроссворд был опубликован в 1875 году в сентябрьском номере журнала «Святой Николас» в Нью-Йорке. При этом первый кроссворд, соответствующий современным представлениям о кроссворде, был создан журналистом Артуром Уинном и опубликован в воскресном номере газеты «New York World» 21 декабря 1913 года [1].

Кроссворды стали популярны в середине 1920-х годов. Один из первых советских кроссвордов («переплетенные слова») был опубликован в номере от 18 августа 1925 года ленинградской «Новой вечерней газеты» [1].

Широкую популярность приобрели кроссворды, публиковавшиеся в течение многих десятилетий в журнале «Огонек» [1].

С прошлого века кроссворды продолжают развиваться как по форме, так и по содержанию. Существует множество разновидностей этой игры. В разных странах есть свои любимые варианты кроссворда, причем они могут использоваться не только как полезное развлечение, но и в учебных целях. Во многих странах проводятся конкурсы по решению и составлению кроссвордов, существуют клубы любителей кроссвордов, к примеру, в России – Международный клуб русских кроссвордов «Крестословица» в Санкт-Петербурге) [1].

Несмотря на то что существует достаточно много разновидностей кроссворда, классический кроссворд не теряет своей популярности и актуальности.

Кроссворды чаще всего располагаются в печатных изделиях, а следствием производства печатных изделий является ущерб природе, так как для получения бумаги необходимо дерево. К тому же, в век цифровизации, все больше людей начинает переходить на электронные версии классических кроссвордов, которые зачастую являются неудобными в использовании, а также содержат достаточно ограниченный словарь понятий.

Именно поэтому появилась необходимость разработать автоматизированную систему составления и разгадывания классического кроссворда, с помощью которой можно было бы конструировать классический кроссворд в ручном или автоматическом режиме, а также выполнять разгадывание кроссворда.

Разработка системы будет производиться по технологии быстрой разработки приложений RAD (Rapid Application Development), которая поддерживается методологией структурного проектирования и включает элементы объектно-ориентированного проектирования и анализа предметной области.

Особенностями данной технологии являются: небольшая команда программистов (от 2 до 4 человек); короткий, но тщательно проработанный производственный график (от 2 до 4 месяцев); повторяющийся цикл, при котором разработчики по мере того, как приложение начинает обретать форму, запрашивают и реализуют в продукте требования, полученные через взаимодействие с заказчиком [2].

Также при разработке системы будет применяться технология ООАП. ООАП (Object-Oriented Analysis/Design) – технология разработки программных систем, в основу которых положена объектно-ориентированная методология представления предметной области в виде объектов, являющихся экземплярами соответствующих классов [3]. Методология

ООАП тесно связана с концепцией автоматизированной разработки программного обеспечения (Computer Aided Software Engineering, CASE) и языком моделирования UML (Unified Modeling Language).

1 Описание и анализ предметной области

Предметная область – часть реального мира, которая имеет существенное значение или непосредственное отношение к процессу функционирования программы, она включает в себя только те объекты и взаимосвязи между ними, которые необходимы для описания требований и условий решения конкретной задачи [3].

Кроссворд, как и многие игры, не имеет строгих правил и жёстких ограничений, но есть традиции, которых придерживается большинство «кроссвордных» изданий. Обычно, когда упоминаются «правила кроссворда», имеется в виду именно этот негласный стандарт, и уточняются только отклонения от него [1].

1.1 Описание предметной области

1.1.1 Правила классического кроссворда

Кроссворд – головоломка, представляющая собой переплетение рядов клеточек, которые заполняются словами по заданным значениям [1].

К каждому слову даётся текстовое определение, в описательной или вопросительной форме указывающее некое слово, являющееся ответом. Ответ вписывается в сетку кроссворда и, благодаря пересечениям с другими словами, облегчает нахождение ответов на другие определения [1].

Загаданные слова представлены в кроссворде в виде цепочки ячеек, в каждую из которых по порядку вписываются буквы ответа – по одной в каждую ячейку. В классическом кроссворде ячейки имеют вид квадратных клеток, собранных в прямую линию [1].

На рисунке 1 приведено изображение примера классического кроссворда.

Слова «пересекаются» друг с другом, образуя сетку кроссворда. Сетка должна быть связной, без изолированных участков, «оторванных» от остальной сетки. Классическая сетка кроссворда состоит из слов, написанных

по вертикали (сверху вниз) и горизонтали (слева направо). Любое слово должно быть пересечено как минимум дважды [1].



Рисунок 1 – Пример классического кроссворда

Для привязки ответов к определениям в кроссворде последовательно нумеруются ячейки, содержащие первые буквы ответов. Нумерация идет по правилам чтения: слева направо и сверху вниз. Слова, идущие из одной клетки в разных направлениях, нумеруются одной цифрой. В списке определений уточняется направление каждого слова (чаще всего определения сгруппированы по направлениям) [1].

Слова-ответы должны быть существительными в именительном падеже и единственном числе. Множественное число допускается только тогда, когда оно обозначает единственный предмет (то, что в лингвистике называется *pluralia tantum*) или единственное число редко употребляется («родители», а не «родитель») [1].

Во многих языках это правило не имеет смысла (так как одно слово может выполнять роль и существительного, и прилагательного, и даже глагола) и не соблюдается [1].

В ответах кроссворда не различаются прописные и строчные буквы. Во многих языках принято не делать различий между определёнными буквами (в частности, опускать диакритические знаки). В русском языке это правило применяется к букве «Ё», приравнивающейся к «Е» [1].

Хорошим тоном (но не правилом) считается симметрия сетки кроссворда относительно вертикальной, горизонтальной или диагональных осей. Возможна также симметрия относительно центральной точки, при которой сетка не изменяется при повороте на 180° [1].

Традиционно ячейка для буквы обозначается белым цветом, а пустое пространство, со всех сторон окруженное белыми ячейками, заливается чёрным или серым цветом. Обычно рамка белой ячейки тоньше на границе двух ячеек, что визуально подчеркивает их объединение [1].

1.1.2 Классификация кроссвордов

«Кроссвордами» в русскоязычных развлекательных газетах зачастую называются головоломки, в которых слова не пересекаются (а это основное правило кроссворда) или слов нет вовсе (как в так называемых «японских кроссвордах»). Очень часто «географическое» название не несёт никакой смысловой нагрузки: «американским кроссвордом» называют головоломку, сочетающую правила классического и «японского кроссворда», при этом в США и Японии действительно есть кроссворды, отличающиеся от европейских, но это все-таки кроссворды, хотя и с несколькими дополнительными правилами [1].

В американском варианте кроссворда все клетки должны находиться на пересечении слов. Так что сетка получается не разреженной, как в европейских, а плотной, как в скандинавских кроссвордах. Правда, составители этих кроссвордов не считают зазорным использовать в качестве загаданных слов аббревиатуры, разговорные или иноязычные слова [1].

Сетка американского варианта кроссворда представлена на рисунке 2.

В японском варианте кроссворда черные клетки не должны соприкасаться сторонами (а значит, не должно быть блоков из черных клеток – соответственно, плотность сетки также приближается к сканвордной) и угловые клетки сетки должны быть белыми (так что сетка обязательно остается строгим прямоугольником). Очевидно, ответы вписываются на

японском и (реже) иероглифами. Поэтому допустимы даже «двухклеточные» слова [1].

1	2	3	4		5	6	7	8	9		10	11	12	13
14					15						16			
17				18						19				
20				21				22						
23			24				25							
		26				27				28		29	30	31
32	33				34				35					
36				37								38		
39			40								41			
42						43				44				
			45		46				47				48	49
50	51	52						53				54		
55							56				57			
58					59						60			
61					62						63			

Рисунок 2 – Сетка американского варианта кроссворда

Сетка японского варианта кроссворда представлена на рисунке 3.

1	2	3			4	5	6	
7				8				
9			10					11
		12			13		14	
15	16				17	18		
19			20	21				
		22		23			24	
	25		26			27		
28					29			

Рисунок 3 – Сетка японского варианта кроссворда

Также довольно популярной головоломкой, встречающейся в печати, является скандинавский кроссворд (сканворд).

Скандинавский кроссворд (сканворд) от классического кроссворда отличается значительно большим количеством пересечений слов по вертикали и горизонтали, а также тем, что в сканворде вместо развернутых вопросов в отдельной графе в отдельных клеточках пишутся краткие определения, по ассоциации с которыми можно угадать искомое слово.

Вопросами в сканворде могут также служить изображения или фотографии – как правило, занимающие несколько клеток сетки либо пронумерованные. В идеале плотность сетки сканворда должна быть стопроцентной. То есть все его поле должно быть заполнено клетками, в которые вписывается либо определение, либо буква отгаданного слова [1].

Скандинавский кроссворд представлен на рисунке 4.

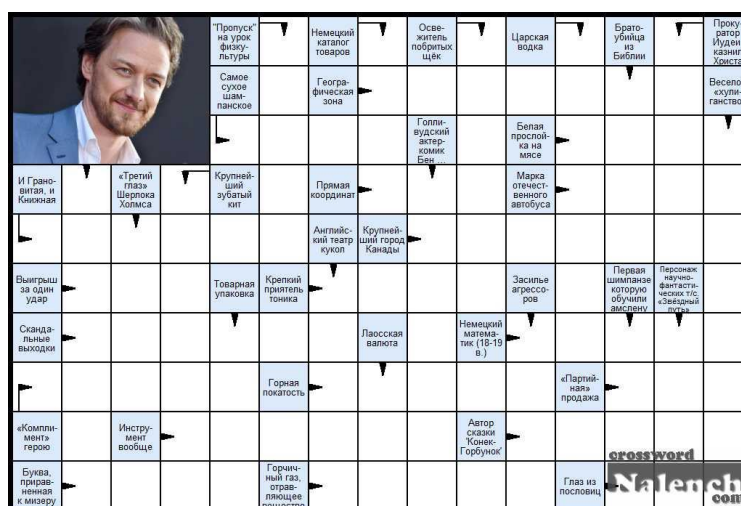


Рисунок 4 – Скандинавский кроссворд

Существует и множество других головоломок, встречающихся в печати, которые пользуются не меньшей популярностью.

1.2 Описание систем-аналогов

В рамках ознакомления с существующими системами по решению и созданию классических кроссвордов с целью их сравнительного анализа, выявления достоинств и недостатков, было рассмотрено несколько источников. Ниже приведено описание некоторых из них.

1.2.1 Система-аналог «Crosswordus»

Одной из систем-аналогов позволяющей работать с классическими кроссвордами – создавать и решать их – является программа Crosswordus [4]. На рисунке 5 приведена главная экранная форма программы «Crosswordus», на которой отображено окно создания кроссворда.

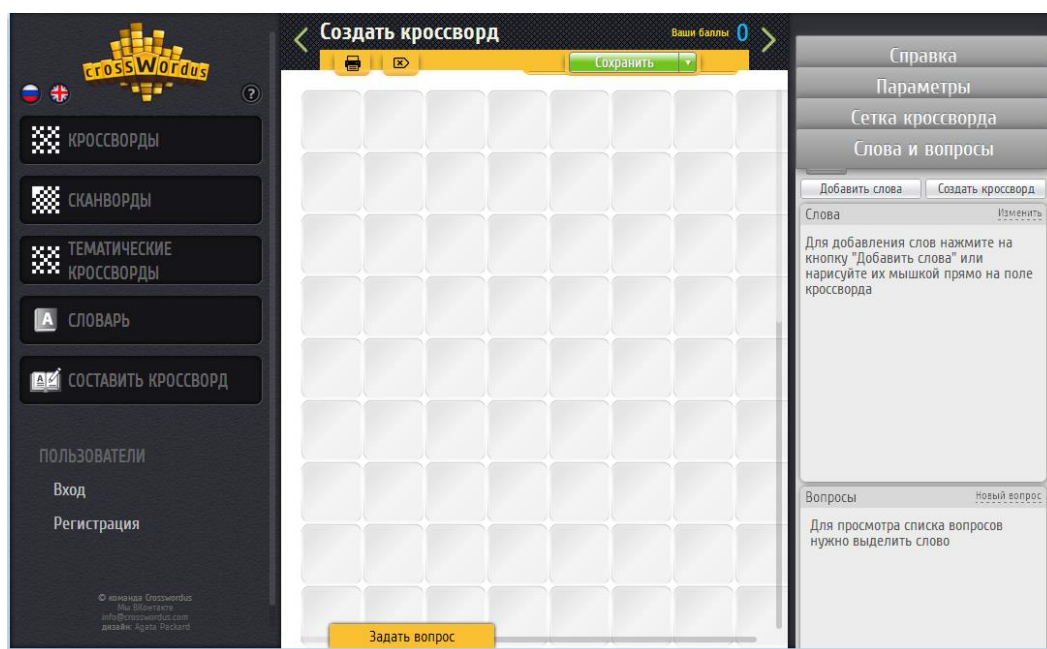


Рисунок 5 – Экранная форма программы «Crosswordus»

К достоинствам данной системы относятся:

- система является бесплатной;
- не требует обязательной регистрации пользователя;
- поддерживает сканворды и тематические кроссворды;
- есть возможность брать определения из словаря;
- имеет возможность сохранения кроссвордов.

К недостаткам системы относятся:

- требуется постоянное подключение к интернету.

1.2.2 Система-аналог «Decalion»

Основное назначение программы «Decalion» – помощь при составлении кроссвордов с последующим оформлением, выводом на печать или экспортом в графический файл. Программа позволяет использовать более чем 70000 слов из 6 входящих в комплект программы словарей: Словарь Ожегова, Толковый словарь русского языка, Морских терминов, Имен, сборный словарь «Брокгауза и Ефрона» и «Большой энциклопедии» и Казахского языка. Программой обеспечивается быстрый поиск необходимых

слов, а также поддерживается возможность создания собственных пользовательских словарей [5].

На рисунке 6 приведена главная экранная форма программы «Decalion», на которой показан процесс составления кроссворда.

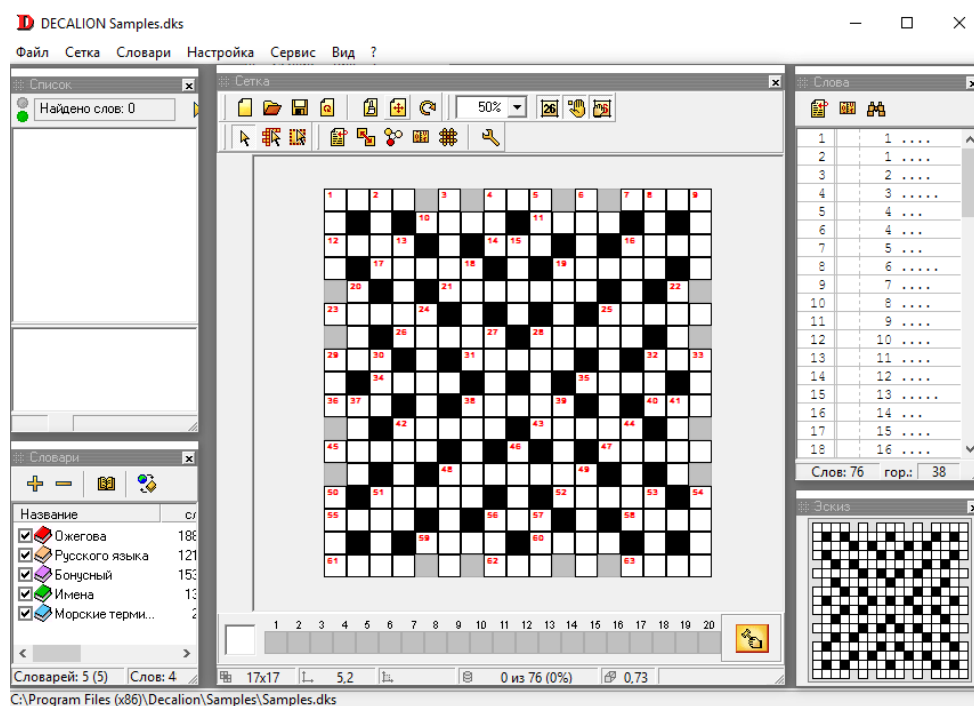


Рисунок 6 – Основная форма программы «Decalion»

К достоинствам данной системы относятся:

- быстрое изменение формы кроссворда;
- мгновенный поиск;
- подсказки программы;
- автоматическая расстановка номеров слов.

К недостаткам системы относятся:

- устаревший дизайн;
- не интуитивно понятный интерфейс.

На основании анализа возможностей систем-аналогов были сформулированы требования к разрабатываемой системе (см. таблицу 1).

Таблица 1 – Сравнительные характеристики систем-аналогов

Название показателя	Система-аналог «Crosswordus»	Система-аналог «Decalion»	Разрабатываемая система
Бесплатность системы	Да	Да	Да
Обязательная регистрация пользователей	Нет	Нет	Да
Поддержка различных видов кроссвордов	Да	Нет	Нет
Сохранение кроссворда	Да	Да	Да
Создание собственных кроссвордов	Да	Да	Да
Создание собственных пользовательских словарей	Нет	Да	Нет
Постоянное подключение к сети Интернет	Да	Нет	Нет
Современный интерфейс	Да	Нет	Да

1.3 Диаграмма объектов предметной области

В основе объектно-ориентированного подхода лежит объектная декомпозиция, при этом статическая структура системы описывается в терминах объектов и связей между ними, а поведение системы – в терминах

обмена сообщениями между объектами. Каждый объект системы обладает своим собственным поведением, моделирующим поведение объекта реального мира [6].

Объектная декомпозиция дает возможность создавать программные системы меньшего размера путем использования общих механизмов, обеспечивающих необходимую экономию выразительных средств. Использование ООП существенно повышает уровень унификации разработки и пригодность для повторного использования не только ПО, но и проектов, что в конце концов ведет к сборочному созданию ПО. Системы зачастую получаются более компактными, чем их не объектно-ориентированные эквиваленты, что означает не только уменьшение объема программного кода, но и удешевление проекта за счет использования предыдущих разработок. А также уменьшается риск создания сложных систем [7].

На рисунке 7 приведена диаграмма объектов предметной области.

Классический кроссворд состоит из сетки, на которой будут располагаться отгадываемые слова, а также задания, которое состоит из определения. Данное определение разъясняет смысл загаданного слова. Определения слов хранятся в словаре понятий, а сами кроссворды можно сохранять в файл.

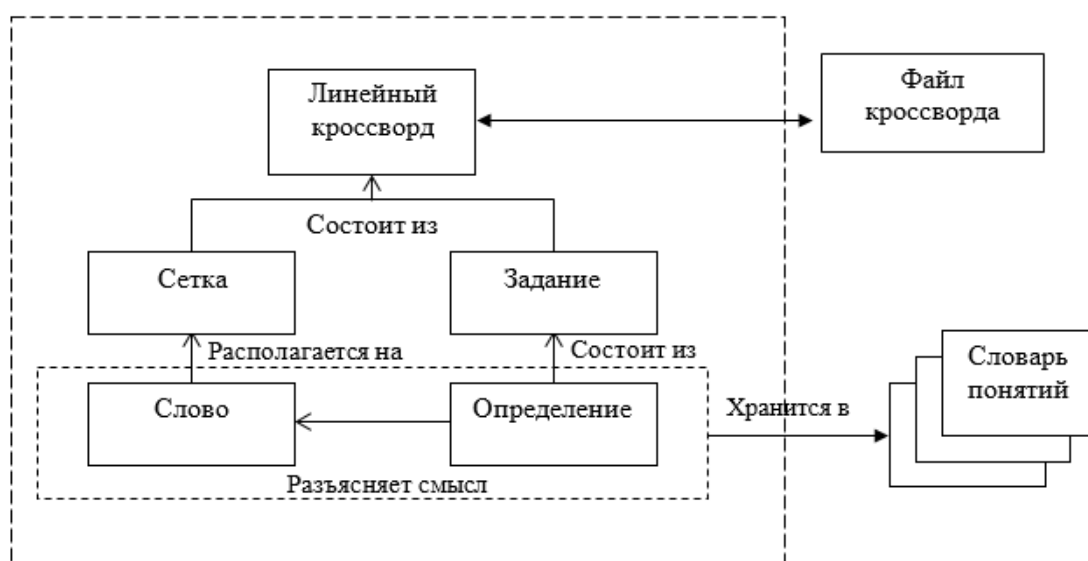


Рисунок 7 – Диаграмма объектов предметной области

1.4 Постановка задачи

Поставленной задачей является автоматизировать систему составления и разгадывания классического кроссворда по выбранной теме. Система должна быть реализована в виде настольного приложения.

В системе должно быть реализовано две роли: администратор и пользователь (игрок). Пользователь должен зарегистрироваться в системе введя логин (от 2 до 8 символов) и пароль (от 4 до 12 символов), а затем авторизоваться на основе введенных регистрационных данных. Система должна проверить введенные учетные данные и настроить интерфейс приложения на заданную роль.

Режим администрирования

Кроссворд должен создаваться на основе следующих параметров, а именно: необходимо указать размерность сетки как по горизонтали (от 15 до 30 клеток), так и по вертикали (от 10 до 25 клеток), подключить словарь понятий. Словарь понятий должен храниться во внешнем файле. Каждое слово и его определение в словаре понятий должны находиться на одной строке и разделяться одним пробелом. Русский язык будет единственным языком записи понятий.

Администратору должна быть доступна возможность создания кроссворда как в ручном, так и в автоматическом режиме.

В автоматическом режиме система должна сгенерировать кроссворд, максимально заполнив сетку.

В ручном режиме создания/редактирования кроссворда администратор должен иметь возможность добавлять/удалять слова, выбирая их из списка доступных слов, загруженных из словаря, а также изменять ориентацию слова. При этом система должна осуществлять фильтрацию словаря понятий по маске. Также у администратора должна быть возможность сохранения кроссворда в файл заданной структуры и последующей загрузки его из файла для редактирования.

Другой задачей администратора будет являться работа со словарем понятий. Администратору должны быть доступны функции добавления/удаления/изменения понятия, сортировка словаря по выбранному критерию (по алфавиту и по длине слова), поиск по заданной маске, а также сохранение словаря в файл и последующая загрузка из файла. Система должна проверять дублирование понятий (понятия должны быть уникальными) и язык записи понятий.

Режим разгадывания

Функциями пользователя будут являться: загрузка кроссворда из файла для последующего разгадывания. Процесс разгадывания кроссворда будет заключаться в выборе слова, вписывании, удалении или изменении буквы. На любом этапе разгадывания должна быть возможность сохранения кроссворда в файл.

Справочная информация

Также в системе должна быть реализована функция выдачи справочной информации, которая будет содержать сведения о системе и ее разработчиках.

Таким образом, система должна решать следующие задачи:

1) общесистемные функции:

- авторизация пользователя в системе (ввод логина, пароля);
- регистрация пользователя в системе (ввод логина, пароля);
- аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
- визуализация процессов работы с кроссвордом;
- автоматическое составление кроссворда по заданным параметрам;
- проверка языка записи понятий;
- сортировка словаря по выбранному критерию;
- ручное составление кроссворда по заданным параметрам;
- фильтрация словаря понятий по маске;

- проверка правильности разгадывания кроссворда;
- выдача справочной информации о системе;

2) функции администратора:

- настройка параметров кроссворда при создании:
 - 1 задание размерности сетки;
 - 2 задание названия кроссворда;
 - 3 подключение словаря понятий;
 - 4 выбор режима составления кроссворда;
- составление/редактирование кроссворда:
 - 1 добавление слова;
 - 2 удаление слова;
 - 3 задание маски для поиска;
 - 4 изменение ориентации слова;
- сортировка словаря по алфавиту;
- сохранение кроссворда в файл заданной структуры;
- загрузка кроссворда из файла;
- работа со словарями понятий:
 - 1 добавление понятия;
 - 2 удаление понятия;
 - 3 изменение понятия;
 - 4 выбор критерия сортировки;
 - 5 поиск по заданной маске;
 - 6 загрузка словаря из файла;
 - 7 сохранение словаря из файла;
 - 8 создание нового словаря понятий;

3) функции пользователя (игрока):

- загрузка кроссворда из файла;
- разгадывание кроссворда:
 - 1 выбор слова;
 - 2 вписывание/удаление/изменение буквы;
- сохранение кроссворда в файл.

2 Проектирование системы

2.1 Структурная схема системы

Сущность структурного подхода к разработке информационной системы заключается в ее декомпозиции (разбиении) на автоматизируемые функции: система разбивается на функциональные подсистемы, которые в свою очередь делятся на подфункции, подразделяемые на задачи и так далее. Процесс разбиения продолжается вплоть до конкретных процедур. При этом автоматизируемая система сохраняет целостное представление, в котором все составляющие компоненты взаимосвязаны [8].

На рисунке 8 приведена структурная схема системы, в ее состав входят следующие подсистемы:

- подсистема авторизации, которая отвечает за выбор роли пользователя или администратора;
- подсистема регистрации, которая отвечает за регистрацию новых пользователей;
- подсистема взаимодействия с базой данных, которая обеспечивает доступ к данным о пользователях;
- файловая подсистема, которая отвечает за сохранение и загрузку кроссвордов;
- подсистема визуализации, которая отвечает за отображение кроссворда при создании и разгадывании;
- подсистема «Игрок», в состав которой входит подсистема разгадывания кроссворда, которая отвечает за вписывание правильных ответов;
- подсистема «Администратор», в состав которой входят:
 - 1 подсистема работы со словарем, которая отвечает за добавление/редактирование/удаление понятий;

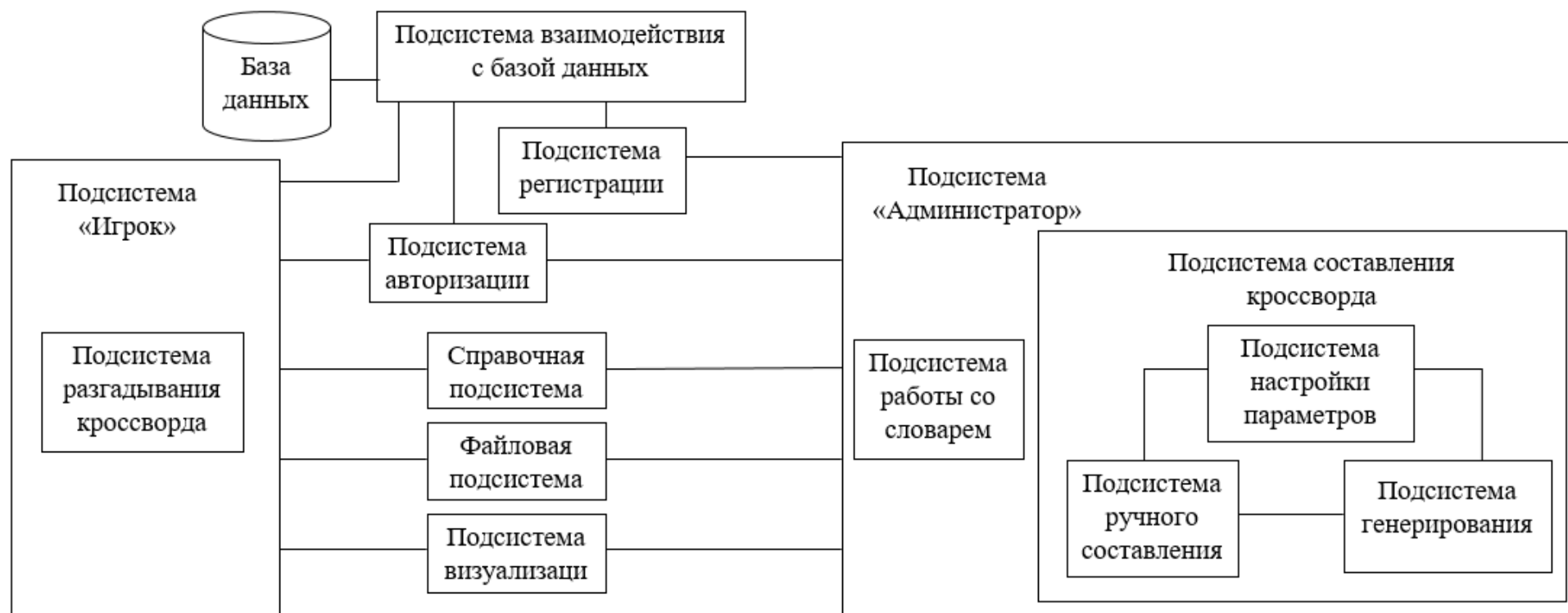


Рисунок 8 – Структурная схема системы

2 подсистема составления кроссворда, в состав которой входит:

- подсистема настройки параметров, которая отвечает за ввод (выбор) значений параметров кроссворда и проверку корректности этих значений;
- подсистема ручного составления, которая отвечает за составление кроссворда в ручном режиме;
- подсистема автоматического генерирования кроссворда, которая позволяет генерировать кроссворд в соответствии с заданными параметрами;
- справочная подсистема, которая отвечает за выдачу справочной информации о возможностях системы и сведений о разработчиках.

2.2 Спецификация системы

Разработка спецификации программного обеспечения является одним из фундаментальных процессов технологии разработки ПО. Этот процесс анализа, формирования, документирования и проверки функциональных возможностей и ограничений системы называется в терминологии программной инженерии «разработка требований» (спецификация требований) [2].

Требования – это свойства, которыми должно обладать ПО для адекватного определения функций, условий и ограничений выполнения ПО, а также объемов данных, технического обеспечения и среды функционирования [9, 10].

На практике часто применяется подход, используемый в различных методологиях разработки ПО и базирующийся на определении групп требований к продукту.

Рассмотрим основные категории требований:

- Программные требования (Software Requirements) – свойства программного обеспечения, которые должны быть надлежащим образом представлены в нём для решения конкретных практических задач [2].

- Функциональные требования задают «что» система должна делать [22].

- Нефункциональные требования – с соблюдением «каких условий» [22].

- Пользовательские требования (User Requirements) – описывают цели/задачи пользователей системы, которые должны достигаться/выполняться пользователями при помощи создаваемой программной системы [2].

Спецификация требований к ПО (SRS) – процесс формализованного описания функциональных и нефункциональных требований, требований к характеристикам качества в соответствии со стандартом качества ISO/IEC 9126-94, которые будут отрабатываться на этапах ЖЦ ПО [2].

В спецификации требований отражается:

- структура ПО;
- требования к функциям, качеству и документации;
- задается в общих чертах архитектура системы и ПО, алгоритмы, логика управления и структуры данных.

Все заинтересованные лица должны прийти к соглашению о том, какая система должна быть создана. Данные соглашения фиксируются в документе, с которым могут сверяться и на который могут ссылаться все участники проекта, - интегрированная спецификация требований, созданная на основе совместного использования традиционной функциональной спецификации и спецификации качества системы [2].

2.2.1 Функциональная спецификация

Функциональная спецификация включает в себя перечень функций системы с привязкой их к конкретной подсистеме и к информационной среде (входные и выходные данные), а также перечень исключительных ситуаций и реакцию системы на их возникновение (при необходимости приводится

перечень ошибок, которые могут возникать в системе и соответствующие им системные сообщения) [2].

Функциональная спецификация системы приведена в таблицах 2, 3, 4.

2.2.2 Спецификация качества

Под качеством (quality) программ понимается то, насколько они соответствуют установленным для них требованиям - спецификациям и сколько высоко установлена планка этих требований, это совокупность черт и характеристик ПС, которые влияют на ее способность удовлетворять заданные потребности пользователей [10].

При разработке системы должны быть обеспечены все показатели качества, в том числе:

1 Устойчивость – свойство, характеризующее способность ПС продолжать корректное функционирование, несмотря на задание неправильных (ошибочных) входных данных.

2 Защищенность – свойство, характеризующее способность ПС противостоять преднамеренным или нечаянным разрушающим действиям пользователя, а также обработка исключительных ситуаций.

3 Временная эффективность – мера, характеризующая способность ПС выполнять возложенные на него функции в течение определенного отрезка времени.

2.2.3 Перечень исключительных ситуаций

Исключительная ситуация – это ситуация, при которой система не может выполнить возложенных на нее функций или которая может привести к денормализации работы системы.

В таблице 5 приведен перечень исключительных ситуаций для разрабатываемой системы и описаны реакции системы на их возникновение.

В таблице 6 приведен перечень сообщений при контроле ввода.

Таблица 2 – Перечень функций, выполняемых системой

Название подсистемы	Название функции	Информационная среда			
		Входные данные		Выходные данные	
		Назначение (наименование)	Тип, ограничения	Назначение (наименование)	Тип, ограничения
1	2	3	4	5	6
1 Подсистема регистрации	1.1 Ввести логин пользователя	Допустимая длина	Целое 2..8	Имя пользователя	Строка
		Допустимый набор символов	Латинский буквы		
	1.2 Ввести пароль	Набор символов	Строка, 4..12 символов	Пароль1	Строка, 4..12 символов
			Латиница + цифры		
	1.3 Повторить ввод пароля	Набор символов	Строка	Пароль2	Строка, 4..12 символов
			Латиница и цифры		

Продолжение таблицы 2

1	2	3	4	5	6
1 Подсистема регистрации	1.4 Проверить совпадение паролей	Пароль1	Строка	Признак совпадения паролей	Логическое
		Пароль2	Строка		
	1.5 Проверить уникальность имени пользователя	Логин	Строка	Признак совпадения логинов	Логическое
		Список логинов зарегистрированных пользователей	Сущность БД «Пользователь»		
2 Подсистема авторизации	2.1 Ввести логин	Допустимая длина	Целое 2..8	Логин	Строка
		Допустимый набор символов	Латинские буквы		
	2.2 Ввести пароль	Допустимая длина	Целое 4..12	Пароль	Строка
		Допустимый набор символов	Латинские буквы + цифры		

Продолжение таблицы 2

1	2	3	4	5	6
2 Подсистема авторизации	2.3 Проверить учетную запись	Логин	Строка	Роль пользователя	Перечислимое
		Пароль	Строка		
		Список учетных записей	Сущность БД «Пользователь»	Код ошибки	Целое
3 Справочная подсистема	3.1 Открыть руководство пользователя	Файл справки	Текстовый (*.HTML)	Код ошибки	Целое
				Визуальное отображение информации	-
	3.2 Выдать сведения о системе	Сведения о системе	Текст (МЕМО)	Визуальное отображение информации	-
	3.3 Выдать сведения о разработчиках	Сведения о разработчиках системы (ФИО, номер группы)	Текст (МЕМО)	Визуальное отображение информации	-

Продолжение таблицы 2

1	2	3	4	5	6
4 Файловая подсистема	4.1 Сохранить кроссворд в файл	Кроссворд	Объект «Кроссворд	Файл	Структура файла определяется в ходе проектирования
		Имя файла	Строка, *.kros		
	4.2 Загрузить кроссворд из файла	Имя файла	Строка, *.kros	Кроссворд	Объект «Кроссворд»
				Код ошибки	Целое
5 Подсистема визуализации	5.1 Отобразить кроссворд	Кроссворд	Объект «Кроссворд»	Отображение сетки, отображение заданий	—
	5.2 Подсветить слово на сетке	Выбранное слово	Строка	Визуализация отображения слова	—

Таблица 3 – Перечень функций, выполняемых подсистемой «Администратор»

Название подсистемы	Название функции	Информационная среда			
		Входные данные		Выходные данные	
		Назначение (наименование)	Тип, ограничения	Назначение (наименование)	Тип, ограничения
1	2	3	4	5	6
1 Подсистема составления кроссворда	1.1 Выделить область на сетке	Кроссворд	Объект «Кроссворд»	Маска	Строка
	1.2 Выбрать слово для добавления	Список понятий	Динамический массив строк	Выбранное понятие из словаря	Строка
	1.3 Добавить слово в кроссворд	Выбранное понятие из словаря	Строка	Кроссворд	Объект «Кроссворд»
		Область на сетке	Координаты		

Продолжение таблицы 3

1	2	3	4	5	6
1 Подсистема составления кроссворда	1.4 Выбрать слово для удаления	Кроссворд	Объект «Кроссворд»	Выбранное понятие	Строка
		Координаты клетки	Целое/Целое		
	1.5 Удалить слово из кроссворда	Выбранное понятие	Строка	Кроссворд	Объект «Кроссворд»
	1.6 Отфильтровать словарь понятий по маске	Список понятий	Объект «Словарь»	Отфильтрованный список понятий	Объект «Словарь»
		Маска	Строка, 1..15 символов		
	1.7 Отсортировать словарь понятий	Список понятий	Объект «Словарь»	Отсортированный список понятий	Объект «Словарь»
		Вид сортировки	Перечислимый		

Продолжение таблицы 3

1	2	3	4	5	6
1 Подсистема составления кроссворда	1.8 Сгенерировать кроссворд по заданным параметрам	Ширина	Целое	Кроссворд	Объект «Кроссворд»
		Высота	Целое		
		Список понятий и определений	Объект «Словарь»		
2 Подсистема настройки параметров	2.1 Подключить словарь понятий	Имя файла	Строка, *.dict	Список понятий и их определений	Объект «Словарь»
				Код ошибки	Целое
	2.2 Ввести ширину кроссворда	Допустимый диапазон значений	Целое 15..30	Текущая длина кроссворда	Целое
	2.3 Ввести высоту кроссворда	Допустимый диапазон значений	Целое 10..25	Текущая высота кроссворда	Целое

Продолжение таблицы 3

1	2	3	4	5	6
3 Подсистема работы со словарем	3.3 Ввести понятие	Допустимый набор символов	Целое 3..15	Понятие	Строка
			Кириллица		
	3.4 Ввести определение	Допустимый набор символов	Целое 1..100	Определение	Строка
			Кириллица, цифры		
	3.5 Добавить понятие	Список понятий и определений	Объект «Словарь»	Измененный список понятий и определений	Объект «Словарь»
		Понятие	Целое 3..15		
		Определение	Целое 0..100		

Продолжение таблицы 3

1	2	3	4	5	6
3 Подсистема работы со словарем	3.6 Изменить понятие и/или определение	Выбранное понятие и определение	Объект «Словарь»	Измененный список понятий и определений	Объект «Словарь»
		Понятие	Целое 3..15		
		Определение	Целое 0..100		
	Выбрать понятие	Список понятий и определений	Объект «Словарь»	Номер понятия	Целое
	3.7 Удалить понятие	Номер понятия	Целое	Измененный список понятий и определений	Объект «Словарь»
	3.8 Отсортировать словарь по критерию	Список понятий и определений	Объект «Словарь»	Отсортированный список понятий и определений	Объект «Словарь»
	3.9 Отфильтровать словарь понятий по маске	Список понятий и определений	Объект «Словарь»	Отфильтрованный список понятий и определений	Объект «Словарь»
		Маска	Целое 1..15		

Продолжение таблицы 3

1	2	3	4	5	6
3 Подсистема работы со словарем	3.10 Проверить дублирование понятий	Список понятий и определений	Объект «Словарь»	Признак дублирования	Логическое
		Понятие	Строка, 3..20 символов		
	3.11 Проверить язык записи понятий	Список понятий и определений	Объект «Словарь»	Признак языка	Логическое
	3.12 Загрузить словарь из файла	Имя файла	Строка, *.dict	Список понятий и определений	Объект «Словарь»
				Код ошибки	Целое
	3.13 Сохранить словарь в файл	Список понятий и определений	Массив строк	Файл	Текстовый
		Имя файла	Строка, *.dict		

Таблица 4 – Перечень функций, выполняемых подсистемой «Игрок»

Название подсистемы	Название функции	Информационная среда			
		Входные данные		Выходные данные	
		Назначение (наименование)	Тип, ограничения	Назначение (наименование)	Тип, ограничения
1 Подсистема разгадывания кроссворда	1.1 Выбрать задание	Список заданий по горизонтали/вертикали	Список строк	№ слова	Целое
	1.2 Вписать букву	Координаты клетки	Целое	Кроссворд	Объект «Кроссворд»
		Буква	Кириллица		
	1.3 Проверить правильность решения	Кроссворд	Объект «Кроссворд»	Результат разгадывания	Логическое

Таблица 5 – Перечень исключительных ситуаций

Название подсистемы	Название исключительной ситуации	Реакция системы
1 Справочная	1.1 Не возможно открыть файл справки	Выдача сообщения «Файл справки поврежден»
	1.2 Не возможно найти файл справки	Выдача сообщения «Отсутствует файл справки»
2 Файловая	2.1 Попытка открытия файла с нестандартным форматом	Выдача сообщения «Файл поврежден или недопустимого формата»
	2.2 Файл с заданным именем не существует	Выдача сообщения «Файл с данным названием отсутствует»

Таблица 6 – Перечень сообщений при контроле ввода

Название подсистемы	Название исключительной ситуации	Реакция системы
1	2	3
1 Авторизация	1.1 Комбинация имени пользователя и пароля отсутствует	Выдача сообщения «Неверное имя пользователя или пароль»
2 Подсистема работы со словарем понятий	2.1 Нет файла с таким именем	Выдача сообщения «Файл словаря понятий не найден»

Продолжение таблицы 6

1	2	3
2 Подсистема работы со словарем понятий	2.2 Структура файла изменена	Выдача сообщения «Неверная структура файла»
	2.3 Дублирование понятий	Выдача сообщения «В словаре уже содержится данное понятие»
	2.4 Не верно выбран язык	Система не вводит символы, кроме кириллицы

2.3 Разработка прототипа интерфейса пользователя системы

Пользовательский интерфейс – совокупность программных и аппаратных средств, обеспечивающих взаимодействие пользователя с компьютером [11].

Разработка пользовательского интерфейса включает те же основные этапы, что и разработка программного обеспечения [11]:

- постановка задачи – определение типа интерфейса и общих требований к нему;
- анализ требований и определение спецификаций – определение сценариев использования и пользовательской модели интерфейса;
- проектирование – проектирование диалогов и их реализация в виде процессов ввода-вывода;
- реализация – программирование и тестирование интерфейсных процессов.

При проектировании пользовательских интерфейсов необходимо учитывать психофизические особенности человека, связанные с восприятием, запоминанием и обработкой информации [11].

При проектировании пользовательских интерфейсов необходимо учитывать психофизические особенности человека, связанные с восприятием, запоминанием и обработкой информации [11].

Прототип экранной формы «Авторизация» представлен на рисунке 9. Для авторизации в системе пользователь должен будет ввести логин, уникальность которого будет проверяться системой. После этого пользователь должен нажать на кнопку «Вход», система должна будет провести аутентификацию пользователя в соответствии с заданной ролью.

Если пользователь еще не зарегистрирован в системе, то он сможет зарегистрироваться, нажав на кнопку «Регистрация», после чего откроется окно регистрации (рисунок 10). Для регистрации нового пользователя необходимо будет ввести логин, пароль и подтверждение пароля. После чего будет произведено добавление пользователя в базу данных пользователей, а также будет выполнен переход обратно в окно авторизации.

Рисунок 9 – Прототип экранной формы авторизация

Рисунок 10 – Прототип экранной формы регистрация

После авторизации пользователя система должна настроить интерфейс на определенную роль.

2.3.1 Режим игрока

На рисунке 11 приведен прототип экранной формы для разгадывания кроссворда. Для того, чтобы игра началась, пользователь должен будет нажать на кнопку «Начать» и выбрать из ниспадающего меню один из двух вариантов: кнопка «Выбрать кроссворд» дает возможность пользователю выбрать новый кроссворд для разгадывания; кнопка «Загрузить сохраненный» дает возможность пользователю продолжить разгадывать ранее начатый кроссворд.

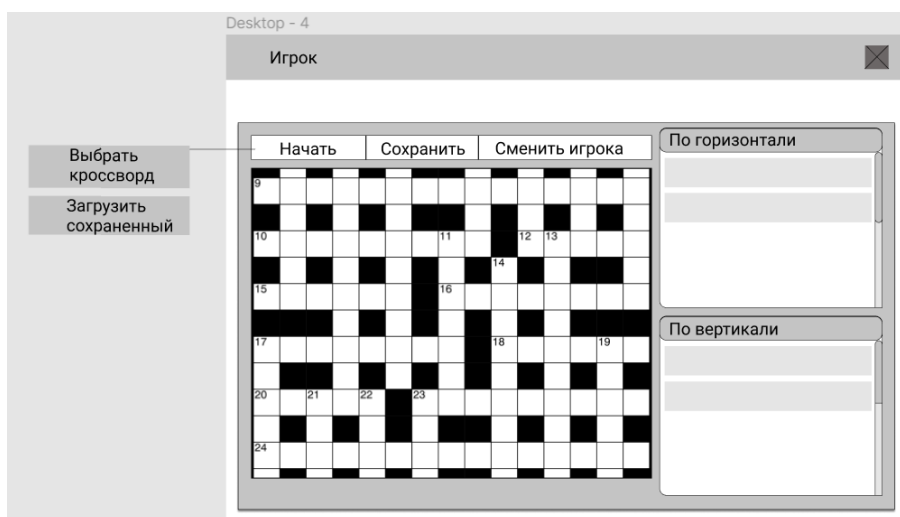


Рисунок 11 – Прототип экранной формы разгадывания кроссворда

После того как пользователь выберет кроссворд, система выведет на экран пустую сетку кроссворда. На панелях «По горизонтали» и «По вертикали» отобразятся определения. Механизм разгадывания следующий: пользователь нажимает на определение из списка «По горизонтали» или «По вертикали», система подсвечивает на сетке строку или столбец, в который нужно вписать понятие, соответствующее выбранному определению. Пользователь заполняет эту строку или столбец. В случае, если пользователь вводит неправильное определение, система подсвечивает его красным цветом и удаляет с сетки. Пользователь может сохранить процесс игры,

нажав на кнопку «Сохранить». В процессе игры у пользователя есть возможность сменить игрока, нажав на кнопку «Сменить игрока».

2.3.2 Режим администрирования

На рисунке 12 приведен прототип экранной формы для создания кроссворда и работы со словарем. Для создания кроссворда пользователь должен будет нажать на кнопку «Создать». Далее откроется экранная форма для создания кроссворда, которая представлена на рисунке 13. Пользователю необходимо будет задать размеры сетки кроссворда, название, загрузить словарь понятий и выбрать способ генерирования кроссворда и нажать на кнопку «Создать».

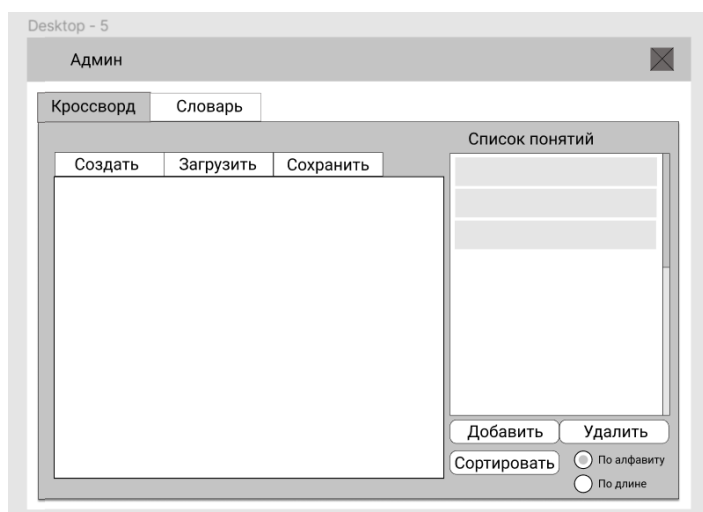


Рисунок 12 – Прототип экранной формы для создания кроссворда и работы со словарем

На рисунке 14 представлен прототип интерфейса работы со словарем понятий. Для работы со словарем пользователь должен нажать на вкладку «Словарь». Пользователь может вручную создать новый словарь понятий, сохранить его в файл или загрузить готовый из файла, нажав на кнопки «Создать», «Загрузить» и «Сохранить» соответственно. Чтобы изменить словарь понятий пользователю нужно будет нажать на слово в области «Понятие» или области «Определение» и ввести новые данные. На лейбле в левом нижнем углу формы система выдаст пользователю количество определений, содержащихся в словаре. Пользователь может выполнить

поиск по маске. Для этого нужно ввести маску в специальное текстовое поле в правом верхнем углу формы и нажать на кнопку с иконкой поиска.

Рисунок 13 – Прототип экранной формы создания кроссворда

Пользователь может осортировать понятия в словаре двумя способами: по алфавиту и по длине. Для этого необходимо выбрать способ, кликнув кружок «По длине» или «По алфавиту» на панели и нажать кнопку «Сортировать».

Рисунок 14 – Прототип экранной формы для создания кроссворда и работы со словарем

На рисунке 15 приведена навигационная модель разрабатываемого приложения.

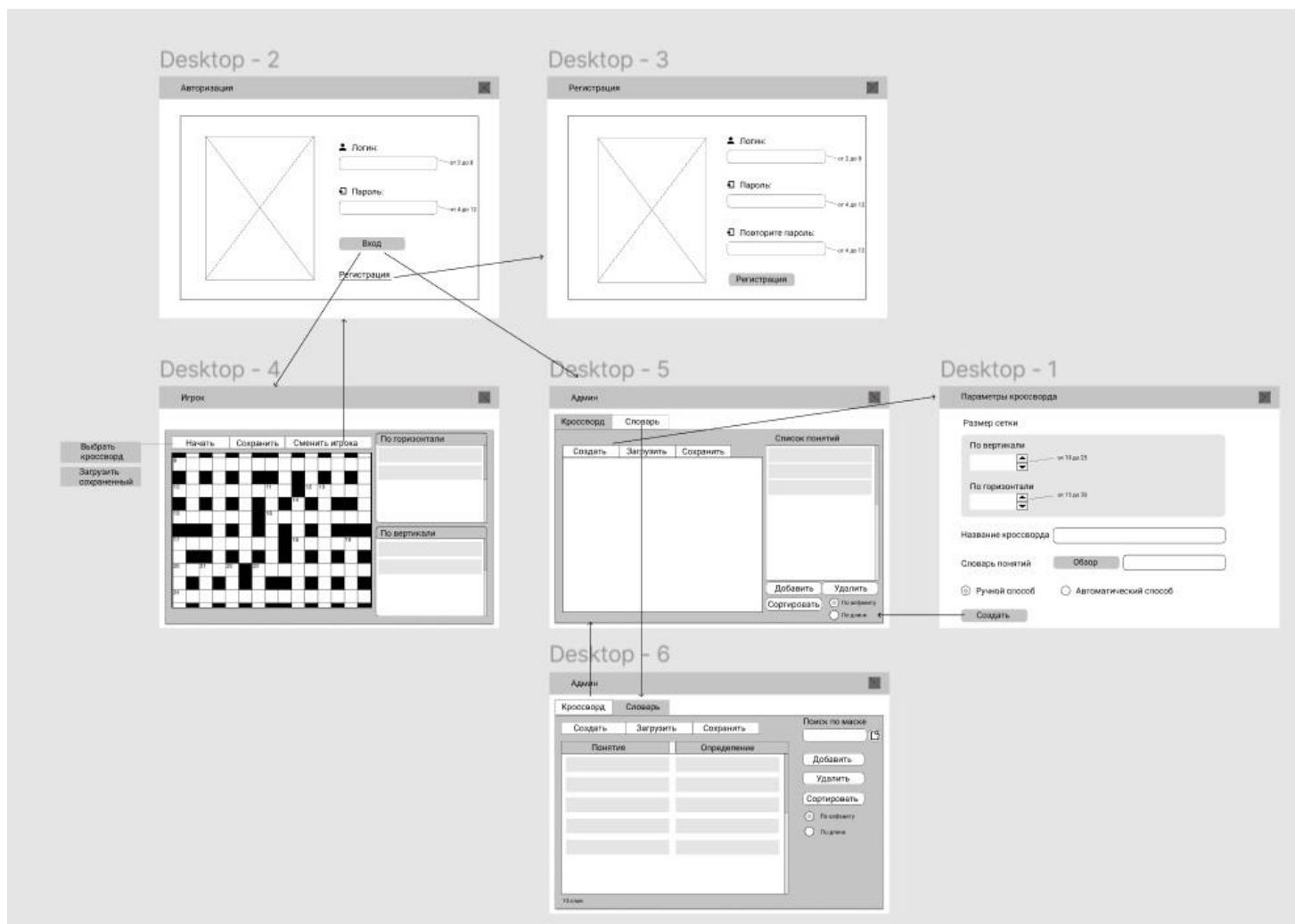


Рисунок 15 – Навигационная модель приложения

2.4 Разработка структур данных и классов

Центральное место в ООАП занимает разработка логической модели системы в виде диаграммы классов. Нотация классов в языке UML проста и интуитивно понятна. Нотация UML предоставляет широкие возможности для отображения дополнительной информации. При этом возможно использование графических изображений для ассоциаций и их специфических свойств [12].

Диаграмма классов (class diagram) служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать, в частности, различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений [12].

Диаграмма классов представляет собой некоторый граф, вершинами которого являются элементы типа "классификатор", которые связаны различными типами структурных отношений. Ее принято считать графически представлением таких структурных взаимосвязей логической модели системы, которые не зависят или инвариантны от времени.

Диаграмма классов состоит из множества элементов, которые в совокупности отражают декларативные знания о предметной области. Эти знания интерпретируются в базовых понятиях языка UML, таких как классы, интерфейсы, отношения между ними и их составляющими компонентами [12].

На рисунке 16 приведена диаграмма классов системы (этап проектирования). В таблице 7 приведено описание классов.

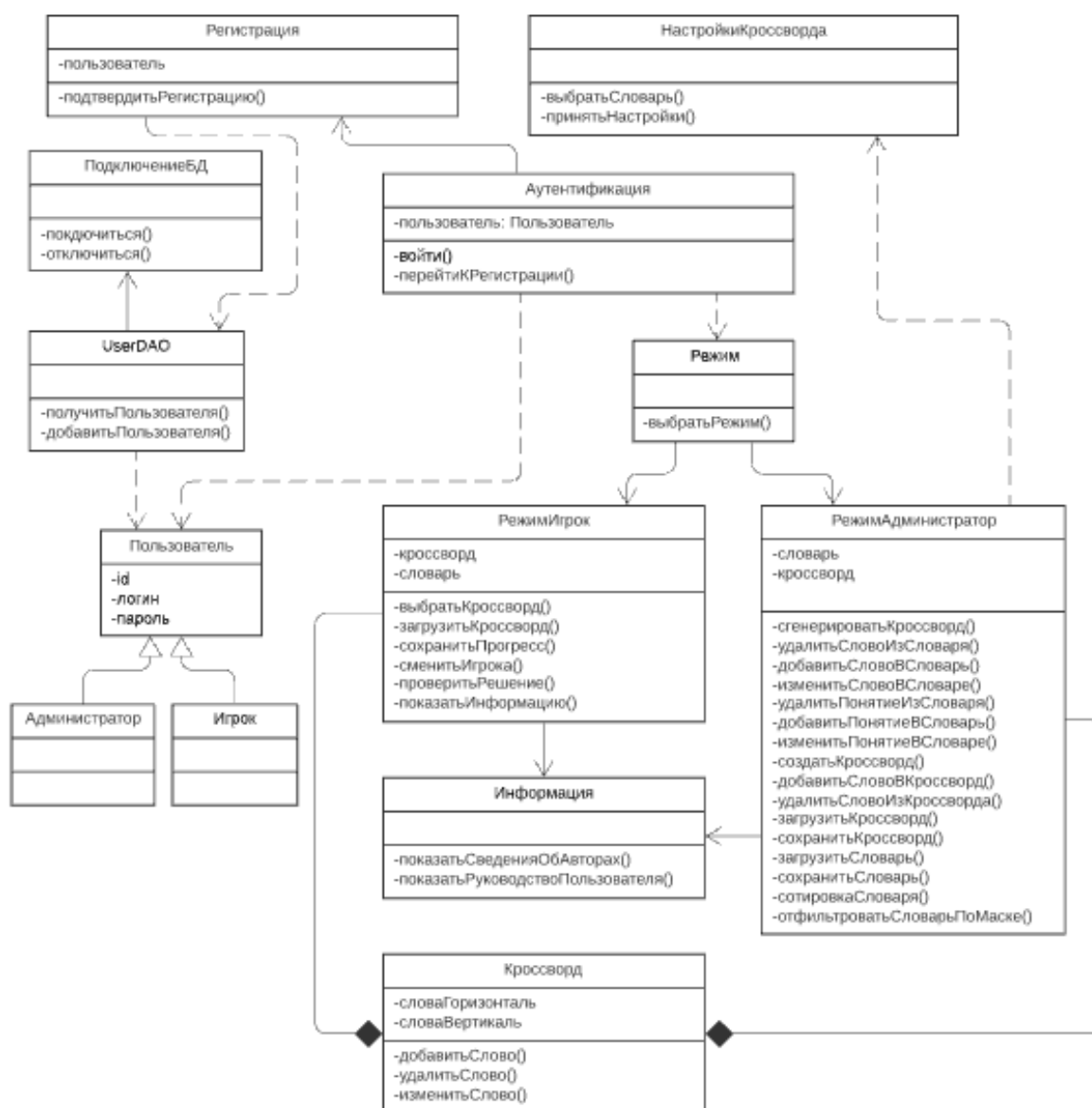


Рисунок 16 – Диаграмма классов

Таблица 7 – Описание классов системы

Название класса	Назначение
1	2
Регистрация	Регистрация нового пользователя в системе.
Аутентификация	Авторизация пользователя в системе.
Информация	Представление сведений об авторах, сведений о системе и руководства пользователя.
НастройкиКроссворда	Настройка параметров кроссворда, выбор словаря понятий.

Продолжение таблицы 7

1	2
РежимАдминистратор	Хранение словаря понятий и кроссворда, их загрузка, создание и изменение, добавление, изменение и удаление слов из словаря понятий и кроссворда, сортировка словаря понятий по длине слова и алфавиту, фильтрация словаря понятий по маске.
РежимИгрок	Хранение словаря понятий и кроссворда, выбор и загрузка кроссворда, проверка решения, смена игрока.
ПодключениеБД	Подключение к базе данных.
UserDAO	Добавление и получение данных пользователя из базы данных.
Пользователь	Модель пользователя
Кроссворд	Модель кроссворда, хранение слов по горизонтали и вертикале, добавление, изменение и удаление слова.

2.5 Логическая модель данных

Проектирование БД является одной из важнейших составных частей процесса создания системы. База данных, рассматриваемая как сложная система, разрабатывается с использованием тех же принципов, что и система в целом. При проектировании баз данных обычно выделяют три уровня абстракции, на которых происходит последовательное уточнение модели: концептуальный (семантический уровень представления данных в виде абстрактных понятий, учитывающих особенности предметной области), логический (уровень представления в виде структуры данных – сущностей, атрибутов и связей) и физический (уровень реализации базы данных) [13].

Логическая информационная модель – модель данных, в которой учитывается способ логического хранения данных в памяти ЭВМ.

На рисунке 17 приведена логическая модель данных системы. В таблице 8 приведено описание сущностей БД.

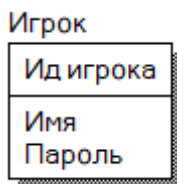


Рисунок 17 – Логическая модель данных системы

Таблица 8 – Сущность «Игрок»

Идентификатор	Тип данных	Описание
Ид игрока	Целый	Уникальный идентификатор пользователя
Имя	Символь- ный[20]	Имя, используемое при идентификации пользователя и его взаимодействии с системой
Пароль	Символь- ный[20]	Пароль пользователя, преобразованный в закодированную строку

2.6 Разработка и описание алгоритмов обработки данных

Алгоритм – конечная совокупность точно заданных правил решения некоторого класса задач или набор инструкций, описывающих порядок действий исполнителя для решения определённой задачи [14].

Если для организации работы системы можно использовать уже известные алгоритмы, то необходимо провести их сравнительный анализ (по эффективности) и выбрать наилучший для данной системы (при введенных ограничениях). В противном случае пользователь разрабатывает свои алгоритмы, обосновывая их необходимость [2].

В данной системе используются следующие алгоритмы:

- диаграмма деятельности администратора при работе с кроссвордом (рисунок 18) и словарем (рисунок 19);

- диаграмма деятельности автоматической генерации кроссворда изображена на рисунке 20.

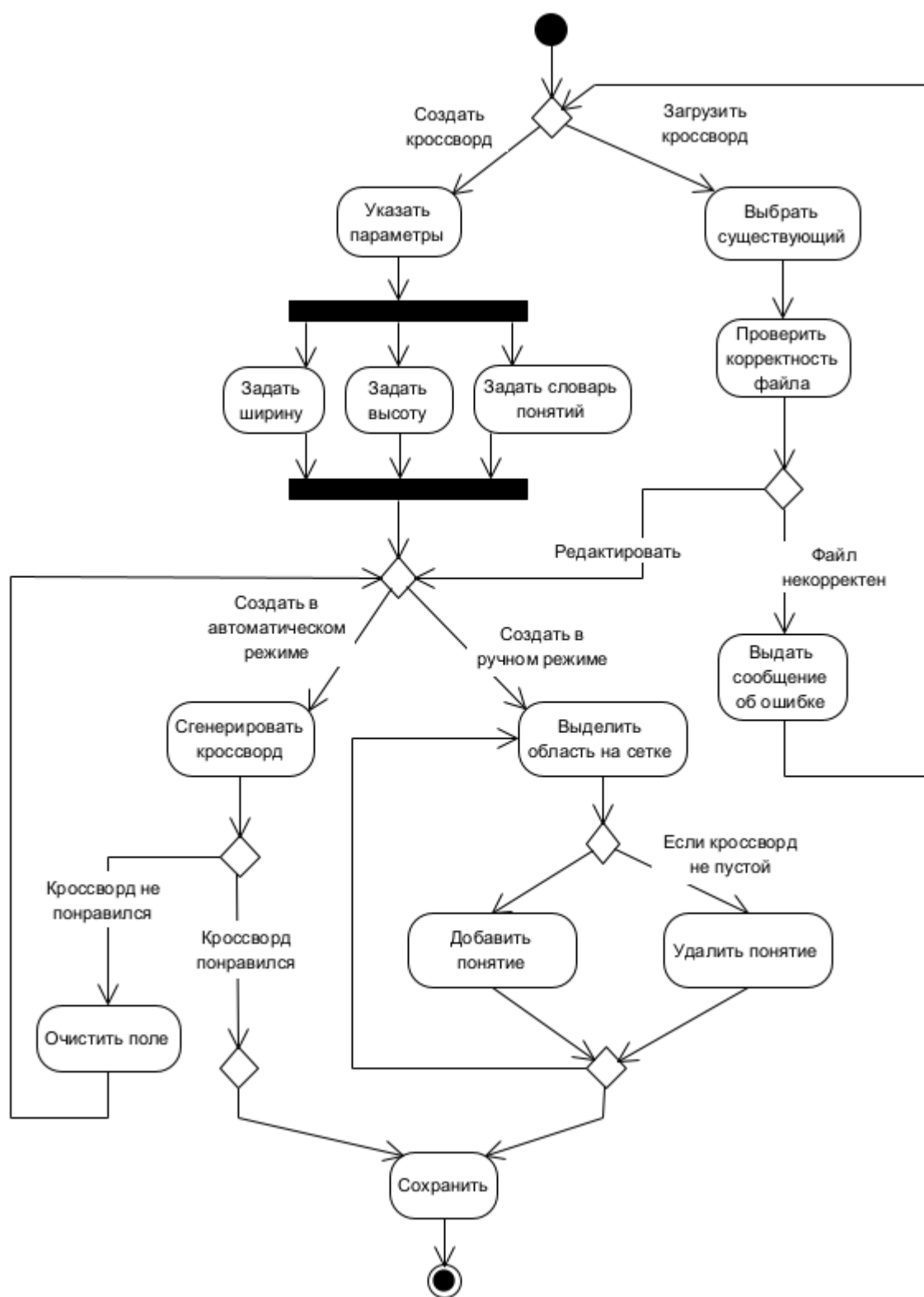


Рисунок 18 – Диаграмма деятельности администратора при работе с кроссвордами

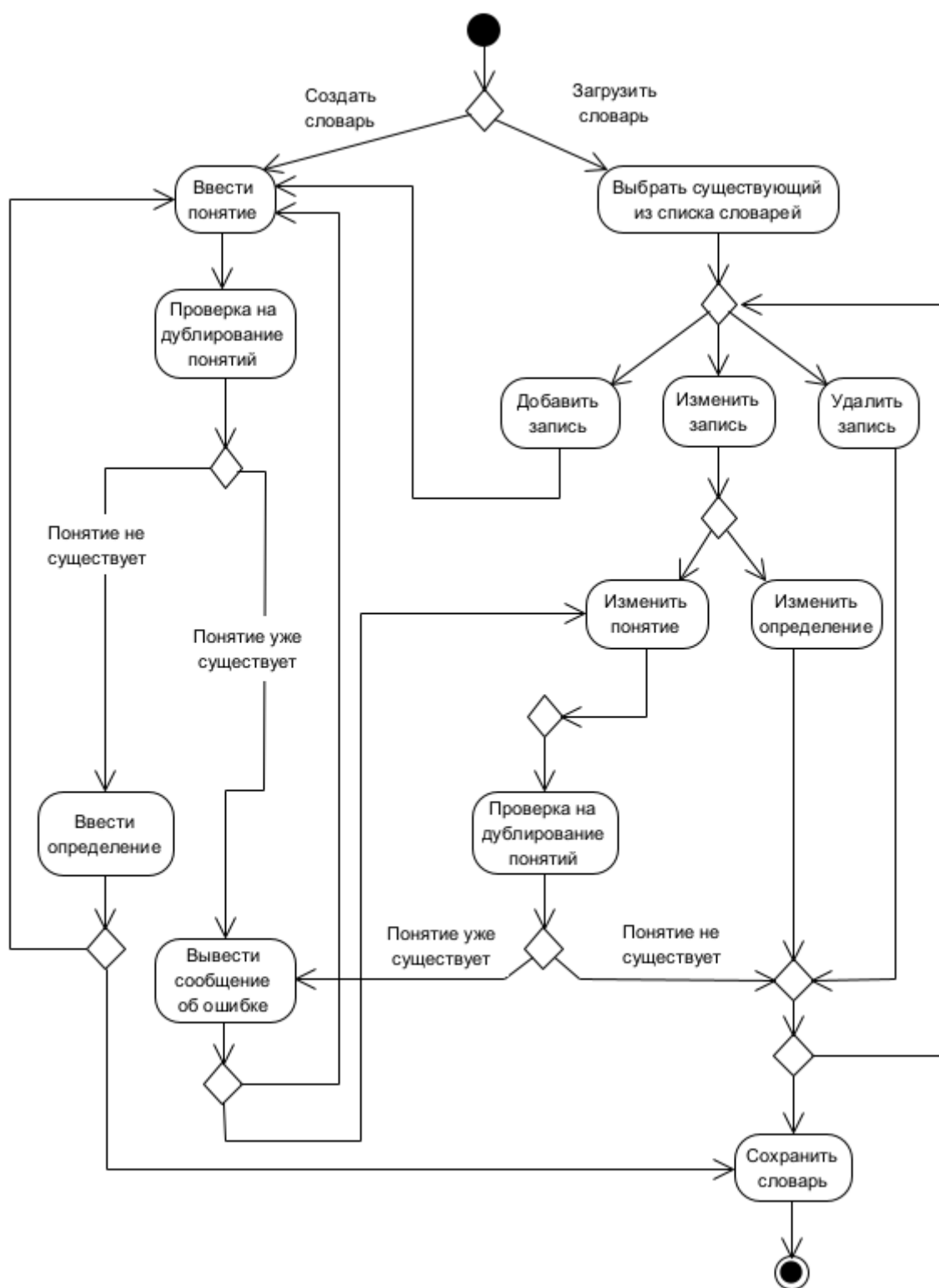


Рисунок 19 – Диаграмма деятельности при работе со словарем

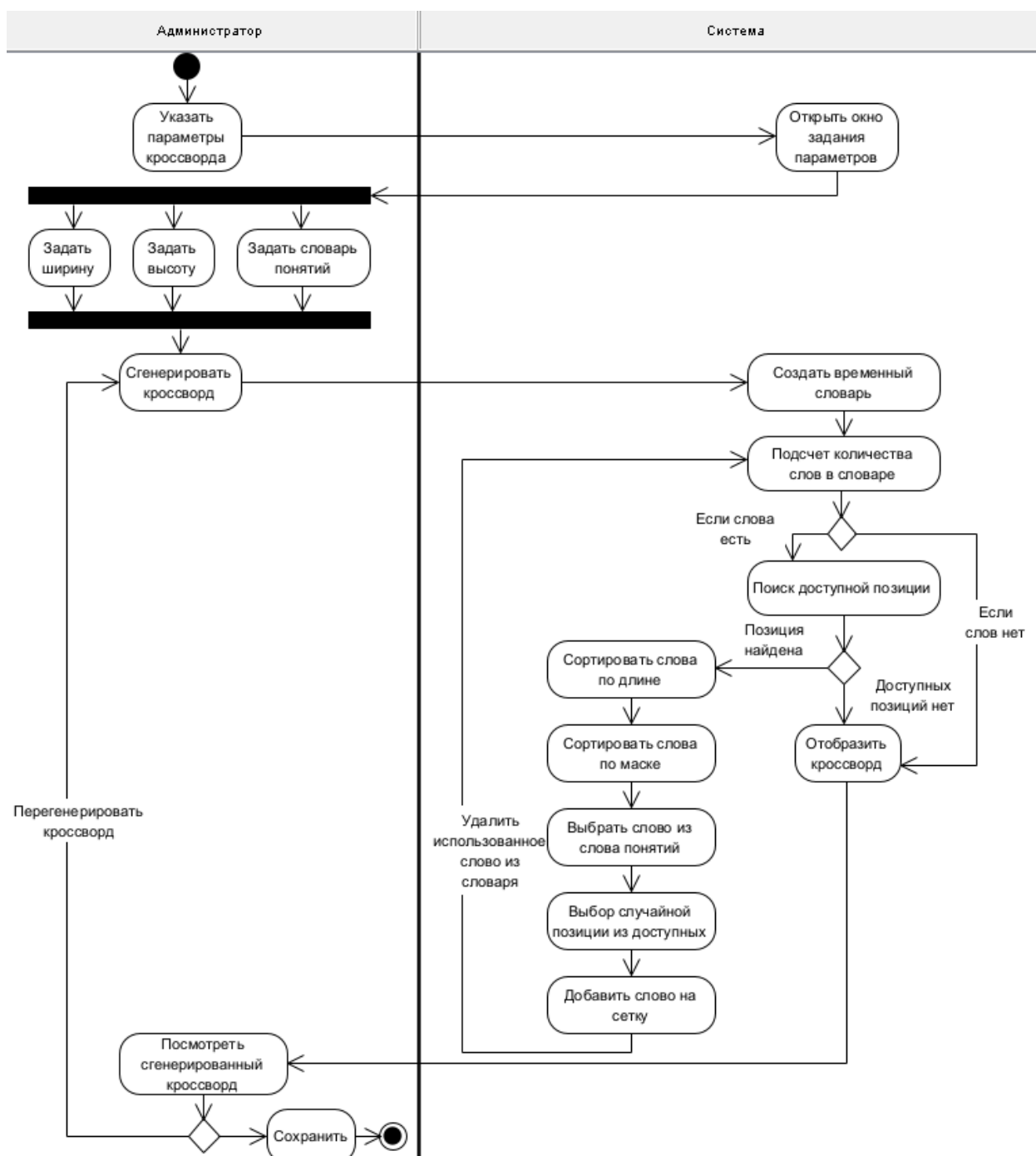


Рисунок 20 – Диаграмма деятельности автоматической генерации кроссворда

2.7 Выбор и обоснование комплекса программных средств

2.7.1 Выбор языка программирования

В качестве языка программирования был выбран язык C#, а в качестве среды разработки Microsoft Visual Studio 2019. Основными преимуществами данного языка являются [15]:

- популярность языка;

- объектная ориентированность – поддержка инкапсуляции, наследования и полиморфизма;

- бесплатность ряда инструментов для небольших компаний и некоторых индивидуальных разработчиков – Visual Studio, облако Azure, Windows Server и др.

- скорость разработки – С# позволяет стартовать разработку быстрее, а это позволяет быстрее получить прототип решения;

- производительность и требовательность к ресурсам – используя С# проще написать код, удовлетворяющий критериям «простоты разработки» и «красоты кода» одновременно;

- библиотеки – множество библиотек с .net идет в базе, плюс к ним множество свободно доступных библиотек, это покрывает практически все первостепенные задачи разработки под Windows;

- удобство отладки;

- язык и синтаксис – синтаксис С# имеет много схожего с другими языками программирования, благодаря чему облегчается переход для программистов. Язык С# часто признают наиболее понятным и подходящим для новичков.

2.7.2 Выбор операционной системы

В качестве операционной системы была выбрана система MS Windows 10. Основными параметрами, которые определили данный выбор, являются [16]:

- 1 Многие программы пишутся специально под эту систему.

- 2 Доступно большое количество качественного бесплатного софта.

- 3 Система является достаточно популярной и официально поддерживаемой со стороны Microsoft.

- 4 Удобный интерфейс, с которым знакомо огромное количество пользователей.

5 В настоящий момент почти каждый пользователь ПК может установить данную систему самостоятельно.

2.7.3 Выбор среды программирования

В качестве среды программирования была выбрана Microsoft Visual Studio.

Microsoft Visual Studio – линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментов. Данные продукты позволяют разрабатывать как консольные приложения, так и игры, и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения и веб-службы [17].

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения [17].

3 Реализация системы

3.1 Разработка и описание интерфейса пользователя

При разработке интерфейса пользователя необходимо учитывать, что интерфейс должен быть не интуитивно, а профессионально понятным пользователю, чтобы пользователь мог быстро разобраться с работой системы и легко взаимодействовать с ней.

При запуске приложения появляется форма «Авторизации». При нажатии на кнопку «Регистрация нового пользователя» система открывает экранную форму для регистрации пользователя. Первым шагом является аутентификация пользователя в системе рисунок 21.

Рисунок 21 – Окно авторизации пользователя

После авторизации пользователя, в системе в зависимости от его роли, открываются разные экранные формы. Если была произведена аутентификация в роли пользователя, то открывается экранная форма для разгадывания кроссворда. Если была произведена аутентификация в роли администратора, то открывается экранная форма для работы с кроссвордом, словарем понятий и учетными записями.

3.1.1 Режим игрока

После авторизации пользователя в системе открывается экранная форма разгадывания кроссворда (рисунок 22).

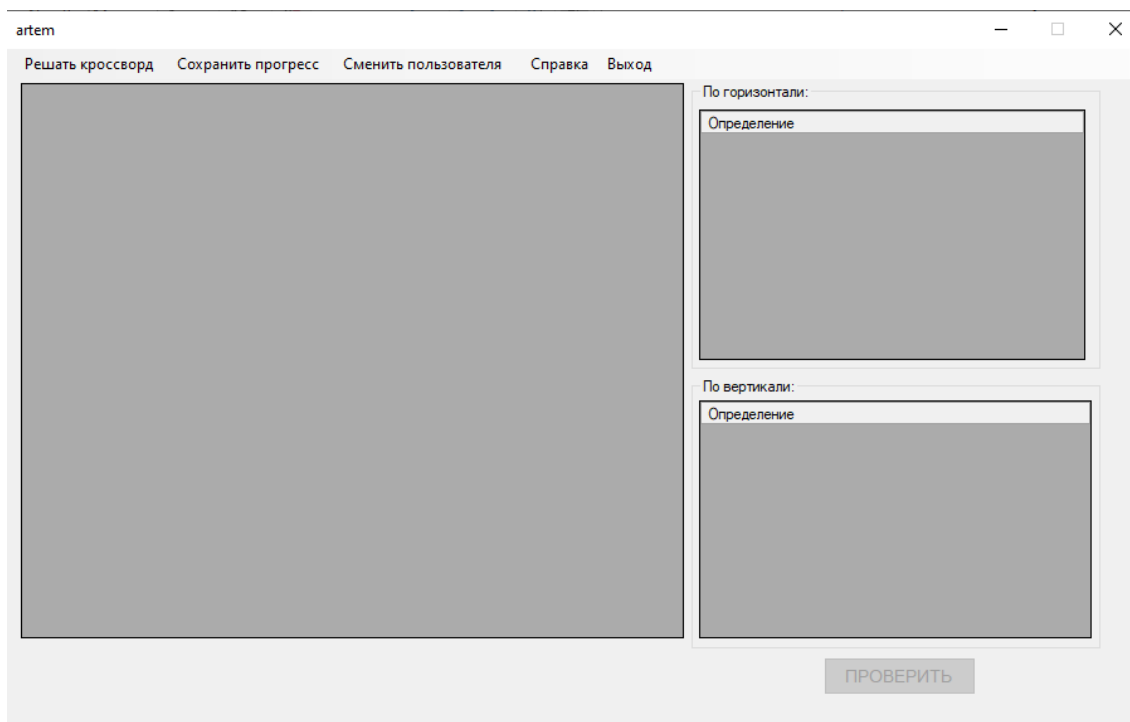


Рисунок 22 – Экранная форма разгадывания кроссворда

В верхнем меню пользователю необходимо выбрать вкладку «Решить кроссворд», а затем загрузить его (рисунок 23).

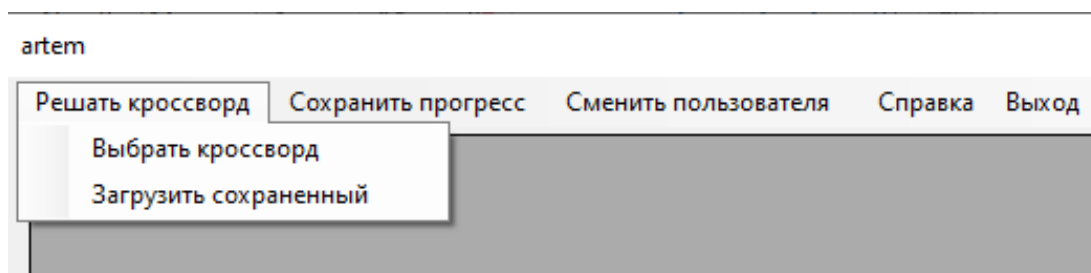


Рисунок 23 – Вкладка загрузки кроссворда

Для того чтобы разгадать слово на сетке, пользователь должен кликнуть на его определение, после чего слово подсветится на сетке (рисунок 24).

Пользователь осуществляет заполнение клеток. После заполнения всех клеток, пользователь нажимает на кнопку проверить, и в случае совпадения всех слов, система выдает ответ, что кроссворд решен верно. В случае хотя бы одного несовпадения, система выдает ответ, что кроссворд решен неверно (рисунок 25).

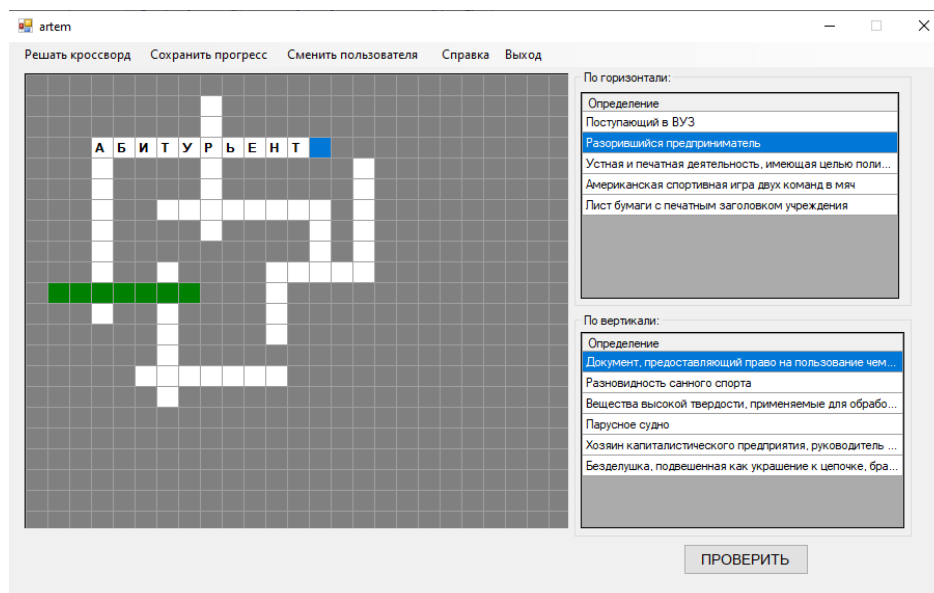


Рисунок 24 – Процесс разгадывания

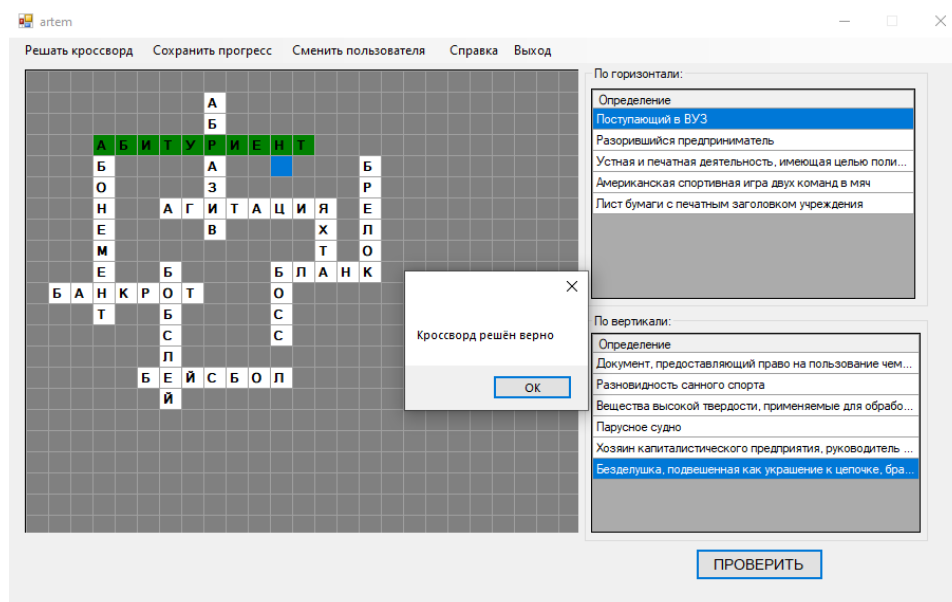


Рисунок 25 – Решенный кроссворд

3.1.2 Режим администратора

Вход в систему в роли администратора происходит с фиксированным логином «admin» и паролем «admin». Экранная форма приложения для роли администратора приведена на рисунке 26. Администратор имеет возможность создавать кроссворд в ручном и автоматическом режиме, загружать уже существующий кроссворд для редактирования, а также сохранять его. Список понятий может быть отсортирован по алфавиту, длине, а также по маске.

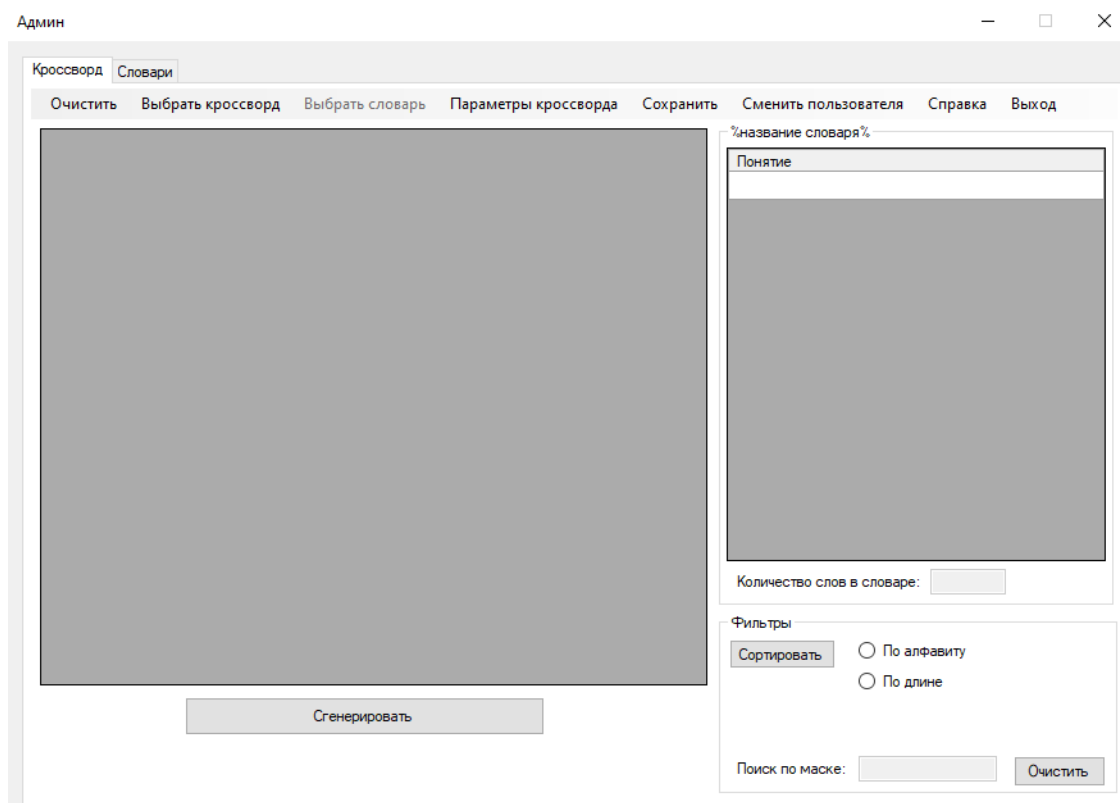


Рисунок 26 – Главная экранная форма администратора

При нажатии на кнопку «Параметры кроссворда» администратор может задать размеры сетки по горизонтали (от 15 до 30 клеток) и вертикали (от 10 до 25 клеток), а также подключить словарь понятий, который хранится во внешнем файле формата .dict (рисунок 27).

При нажатии на кнопку «Выбрать кроссворд» открывается диалоговое окно, где можно выбрать кроссворд (файл формата .Kros), после чего на сетке отобразится загруженный кроссворд.

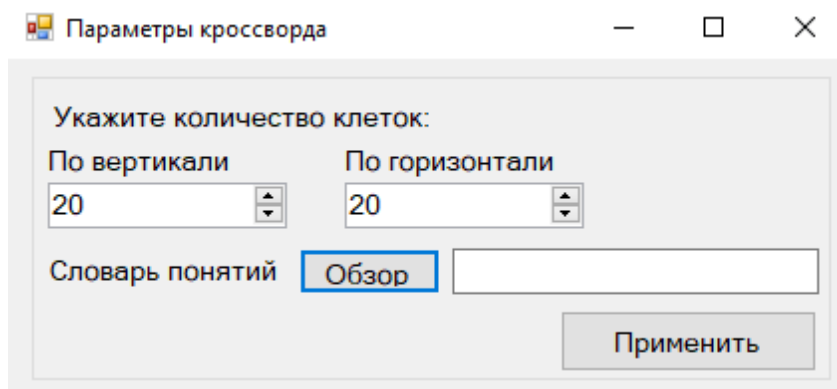


Рисунок 27 – Экранная форма «Параметры кроссворда»

При нажатии на кнопку «Выбрать словарь» открывается диалоговое окно, где можно выбрать словарь (файл формата .dict) после чего в окне словаря понятий отобразится загруженный словарь (рисунок 28).

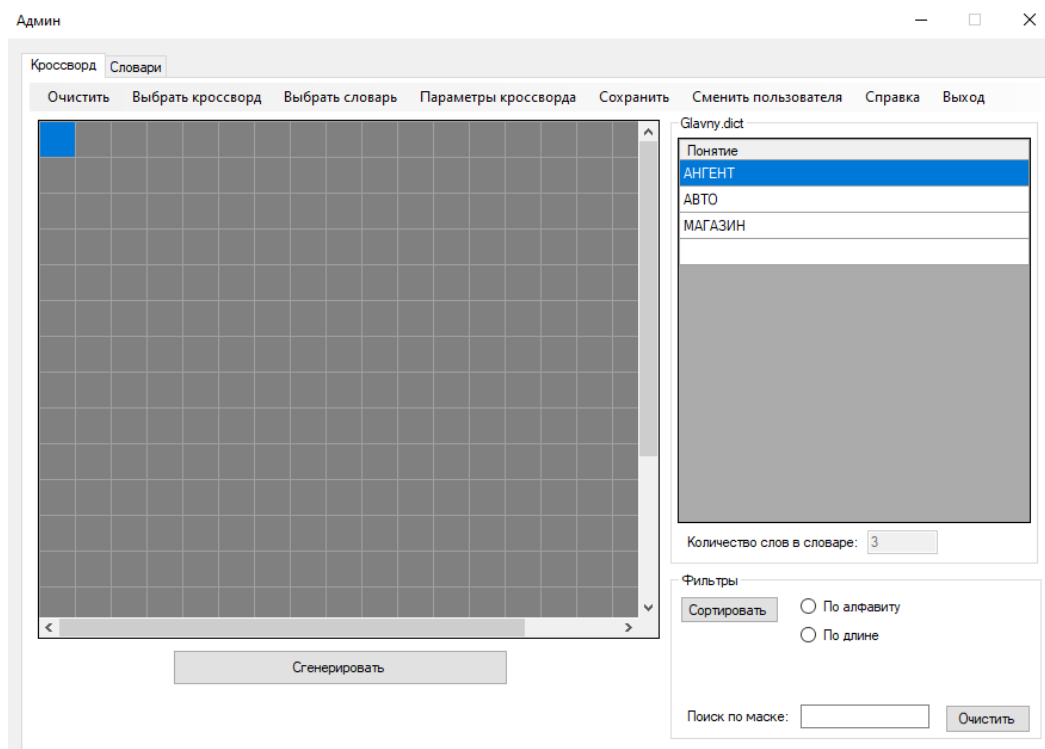


Рисунок 28 – Экранная форма администратора с
загруженным словарем понятий

Словарь понятий может быть отсортирован по алфавиту (рисунок 29) и по длине слов (рисунок 30).

При нажатии на кнопку «Сохранить» открывается диалоговое окно, где можно сохранить кроссворд.

При нажатии на кнопку «Создать» произойдет очистка текущей формы.

При работе с системой в автоматическом режиме достаточно задать размеры сетки в окне параметров кроссворда, указать словарь понятий, после чего нажать на кнопку «Сгенерировать», после которой выполняется генерирование кроссворда и отображение на сетке (рисунок 31).

Glavny.dict

Понятие
АББРЕВИАТУРА
АБЕРРАЦИЯ
АБИТУРИЕНТ
АБЛЯЦИЯ
АБОЛИЦИОНИЗМ
АБОНЕМЕНТ
АБОНЕНТ
АБОРДАЖ
АБОРИГЕН
АБРАЗИВ
АБРАЗИЯ
АБСОЛЮТИЗМ
АБСОРБЦИЯ
АБСТРАКЦИОНИЗМ

Количество слов в словаре: 1000

Фильтры

Сортировать ☒ По алфавиту ☐ По длине

Поиск по маске:

Рисунок 29 – Сортировка словаря по алфавиту

Glavny.dict

Понятие
ФОН
ФАС
ШОК
ШЕФ
ШАХ
ЭРА
БУМ
ШОУ
ФАТ
БОЙ
БАР
БОН
ФЕЯ
БРА

Количество слов в словаре: 1000

Фильтры

Сортировать ☐ По алфавиту ☒ По длине

Поиск по маске:

Рисунок 30 – Сортировка словаря по длине

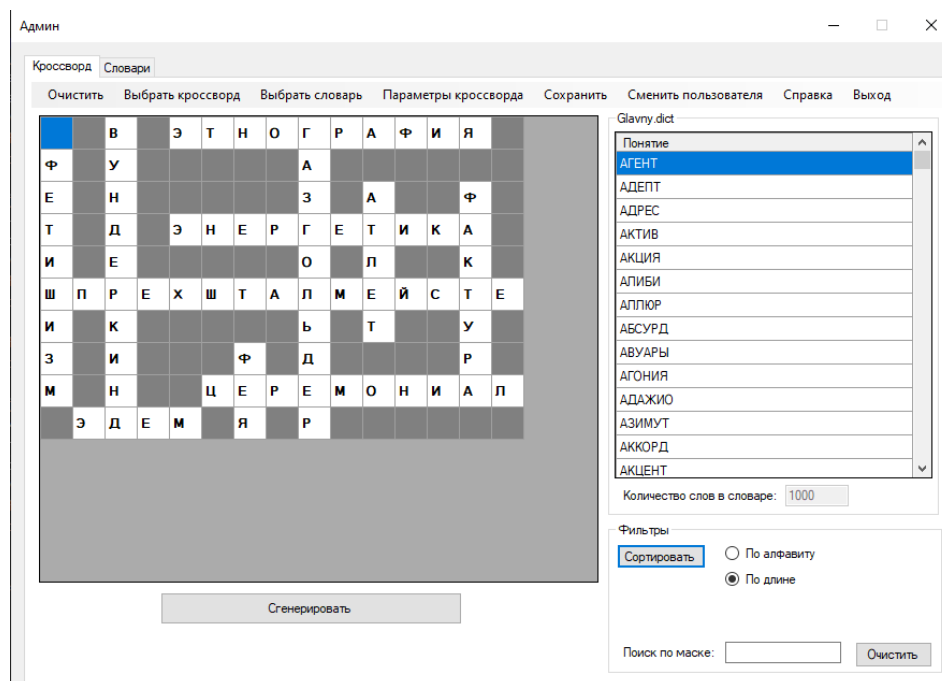


Рисунок 31 – Пример генерации кроссворда в автоматическом режиме

При работе с системой в ручном режиме необходимо выделить на сетке рабочую область, после чего в словаре понятий произойдет отображение слов, наиболее подходящих для данной области (рисунок 32). Для добавления слова на сетку необходимо кликнуть на него два раза.

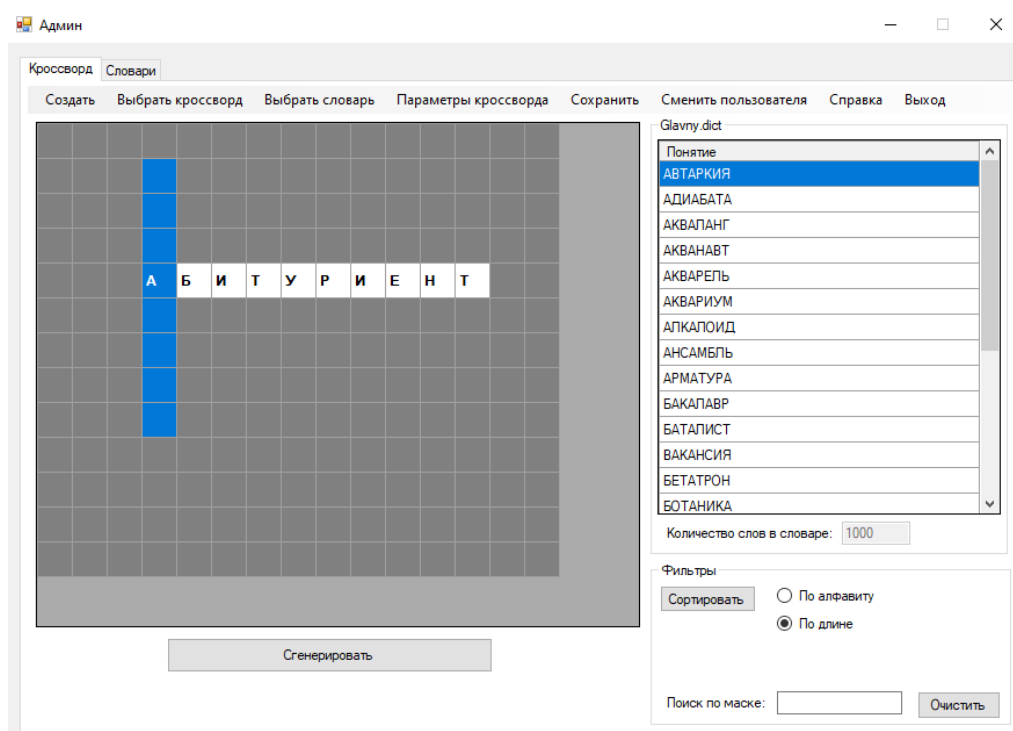


Рисунок 32 – Пример выделения рабочей области для добавления слова

Вкладка «Словари» (рисунок 33) предназначена для работы со словарем понятий. Здесь можно создать, загрузить и сохранить словарь. Также доступны функции добавления, изменения, удаления понятия из словаря, а также сортировки и поиска по маске.

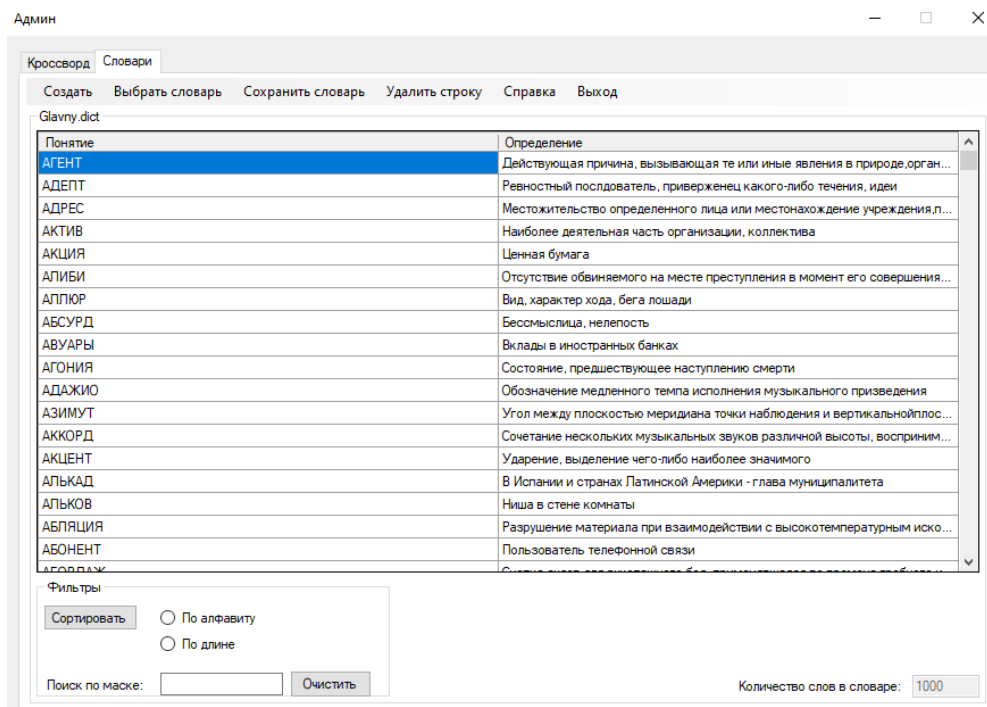


Рисунок 33 – Экранная форма работы со словарем понятий

3.2 Реализация классов и структур данных

В соответствии со спецификацией, приведенной в п. 2.4, и с учетом выбранного языка программирования (см. п. 2.7.1) разработана диаграмма классов системы (этап реализации), приведенная на рисунке 34.

3.3 Реализация и описание модулей программы

Диаграмма модулей (компонентов) описывает особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код. Во многих средах разработки модуль или компонент соответствует файлу. Пунктирные стрелки, соединяющие модули, показывают отношения взаимозависимости,

аналогичные тем, которые имеют место при компиляции исходных текстов программ [18].

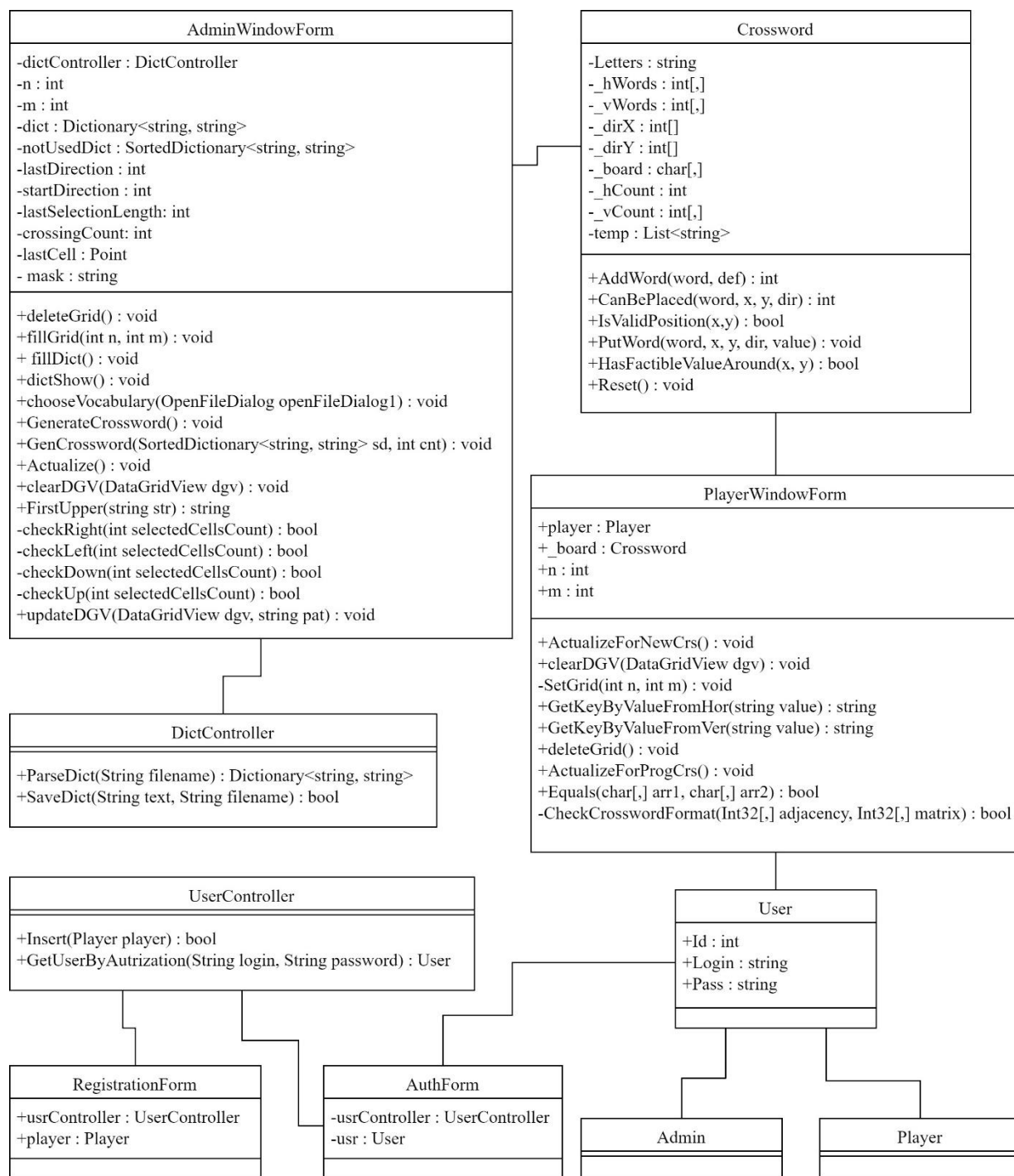


Рисунок 34 – Описание классов системы (этап реализации)

На рисунке 35 приведена диаграмма модулей системы. В таблице 9 приведено описание модулей системы.

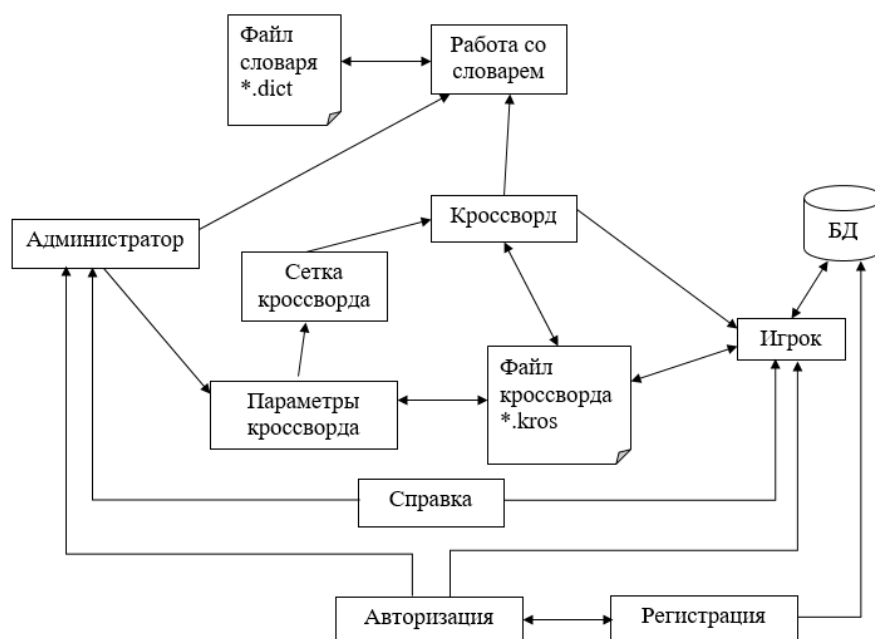


Рисунок 35 – Диаграмма модулей системы

Таблица 9 – Описание модулей системы

Название модуля	Название файла	Назначение	Подсистема
1	2	3	4
Кроссворд	Crossword.cs	Отвечает за генерацию кроссворда	Подсистема составления кроссворда
Авторизация	AuthForm.cs	Отвечает за авторизацию пользователей	Подсистема авторизации
Администратор	Admin.cs	Отвечает за взаимодействие с кроссвордом и словарем понятий	Подсистема Администратор
Игрок	Player.cs	Отвечает за решение кроссворда	Подсистема Игрок

Продолжение таблицы 9

1	2	3	4
Работа со словарем	DictController.cs	Отвечает за редактирование словаря с понятиями и их определениями	Подсистема работы со словарем
Параметры кроссворда	CrosswordSettings.cs	Отвечает за настройку параметров кроссворда	Подсистема настройки параметров
Регистрация	RegistrationForm.cs	Отвечает за регистрацию новых пользователей	Подсистема регистрации
База данных	UserDAO.cs	Отвечает за хранение данных об игроке	Подсистема взаимодействия с базой данных
Справка	AboutAuthors.cs	Отвечает за справочную информацию	Подсистема справки

3.4 Выбор и обоснование комплекса технических средств

3.4.1 Расчет объема занимаемой памяти

Расчет объема внешней памяти

Для расчета необходимого объема свободной внешней памяти, необходимой для функционирования системы, воспользуемся следующей формулой:

$$V_{\text{ЖД}} = V_{\text{ОС}} + V_{\text{ПР}} + V_{\text{БД}} + V_{\text{СПО}} + V_{\text{Ф}},$$

где V_{OC} – объем памяти, занимаемый операционной системой (операционная система Windows 10 Home 64 бит с пакетом обновлений SP1, $V_{OC} = 16$ Гб);

$V_{ПР}$ – объем памяти, занимаемый непосредственно файлами приложения ($V_{ПР} = 40,3$ Мб);

$V_{БД}$ – объем памяти, занимаемый базой данных при ее максимальном заполнении (оценим сверху $V_{БД} = 2$ Кбайт);

$V_{СУБД}$ – объем памяти, занимаемый всем необходимым сопутствующим программным обеспечением ($V_{СУБД} = 2$ Гб);

$V_{Ф}$ – объем памяти, необходимый для хранения файлов, необходимых для работы программы (дадим ему оценку сверху в 520 Кбайт).

Таким образом, суммарный объем внешней памяти составит:

$$V_{ЖД} = 16 \text{ Гб} + 40,3 \text{ Мб} + 2 \text{ Кбайт} + 2 \text{ Гб} + 520 \text{ Кбайт} \approx 18,04 \text{ Гб}.$$

Расчет объема ОЗУ

Для расчета необходимого объема ОЗУ воспользуемся следующей формулой:

$$V_{ОЗУ} = V_{OC} + V_{ПР} + V_{СУБД} + V_{БД},$$

где V_{OC} – ОЗУ, занимаемое операционной системой (2 Гб);

$V_{ПР}$ – ОЗУ, которое займет само приложение (не превысит 20 Мб);

$V_{СУБД}$ – ОЗУ, занимаемое СУБД (оценим его сверху значением в 18 Мб);

$V_{БД}$ – объем данных из базы, который может быть одновременно загружен в оперативную память (дадим ему оценку сверху в 2 Кб).

Суммарные объемы ОЗУ составит:

$$V_{ОЗУ} = 2 \text{ Гб} + 20 \text{ Мб} + 18 \text{ Мб} + 2 \text{ Кб} \approx 2,04 \text{ Гб}.$$

Таким образом, 2 Гб оперативной памяти можно считать минимально необходимым для функционирования системы.

3.4.2 Минимальные требования, предъявляемые к системе

Для корректного функционирования системы необходимо:

- 1) тип ЭВМ: x86-64 совместимый;
- 2) объем ОЗУ – не менее 2 Гб;
- 3) объем свободного дискового пространства – не менее 20 Гб;
- 4) клавиатура или иное устройство ввода;
- 5) мышь или иное манипулирующее устройство.

ЗАКЛЮЧЕНИЕ

Во время лабораторного практикума была разработана автоматизированная система составления и разгадывания классического кроссворда по выбранной теме, позволяющая создавать и разгадывать классический кроссворд.

В первом разделе приведены основные понятия предметной области, исследованы характеристики систем-аналогов, на основании этого выполнена объектная декомпозиция, отраженная в диаграмме объектов. Также сформулирована постановка задачи.

Во втором разделе приведена структурная схема системы, обоснован выбор архитектуры системы, разработана функциональная спецификация, разработаны прототипы экранных форм, диаграмма классов, а также основные алгоритмы работы системы. Кроме того, был выбран комплекс программных средств.

В третьем разделе приведено описание интерфейса полученной системы, реализация классов, диаграмма модулей, а также проведены ресурсные расчеты.

Разработанную систему могут использовать люди, интересующиеся составлением или разгадыванием классических кроссвордов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Кроссворд [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2021. URL: <https://ru.wikipedia.org/wiki/Кроссворд> (дата обращения: 27.09.2021).

2 Зеленко Л.С. Методические указания к лабораторному практикуму по дисциплине «Технологии программирования». Самара: СГАУ, 2012. 69 с.

3 Леоненков, А.В. Нотация и семантика языка UML [Электронный ресурс]/ А.В. Леоненков. – Интернет-университет информационных технологий. URL: <http://www.intuit.ru/department/pl/umlbasics> (дата обращения: 27.09.2021).

4 Сервис CrossWordus [Электронный ресурс] // Сервис CrossWordus – Летние чтения в школе. URL: <https://www.sites.google.com/site/summerreading183/servisy/crosswordus> (дата обращения: 27.09.2021).

5 Программа помощник составления кроссвордов Decalion. [Электронный ресурс] // Decalion – О программе. URL: <http://www.decalion.narod.ru/program.htm> (дата обращения: 27.09.2021).

6 Методология объектно-ориентированного проектирования ИС, объектно-ориентированный подход к проектированию ИС. [Электронный ресурс] // Основные принципы построения объектной модели. URL: https://studref.com/382622/informatika/metodologiya_obektno_orientirovannogo_proektirovaniya (дата обращения: 14.10.2021).

7 Достоинства и недостатки объектно-ориентированного подхода – проектирование информационных систем [Электронный ресурс] // Достоинства и недостатки объектно-ориентированного подхода. URL: https://studref.com/382623/informatika/dostoinstva_nedostatki_obektno_orientirovannogo_podhoda (дата обращения: 14.10.2021).

8 Структурный подход к проектированию ИС [Электронный ресурс] // Сущность структурного подхода. URL: http://citforum.ru/database/case/glava2_1.shtml (дата обращения: 14.10.2021).

- 9 Вигерс К. Разработка требований к программному обеспечению: Пер. с англ. М.: Издательский торговый дом «Русская Редакция», 2004. 576 с.
- 10 Зеленко Л.С. Программная инженерия. Самара: изд-во СГАУ, 2012. 263 с.
- 11 Разработка пользовательских интерфейсов [Электронный ресурс] // Тема 3.2 Разработка пользовательских интерфейсов | Контент-платформа Pandia.ru. URL: <http://pandia.ru/text/78/247/74988.php> (дата обращения: 14.10.2021).
- 12 Диаграмма классов (class diagram) [Электронный ресурс] // Каталог библиотеки кафедры «Информатика и интеллектуальная собственность» URL: <http://khpi-iip.mipk.kharkiv.edu/library/case/leon/gl5/gl5.html> (дата обращения 11.11.2021).
- 13 Построение ER модели данных. Стандарт IDEF1X [Электронный ресурс] // MetodolASOIU – Стр 2. URL: <https://studfile.net/preview/2115111/page:2/> (дата обращения: 11.11.2021).
- 14 Алгоритм [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2021. URL: <https://ru.wikipedia.org/wiki/Алгоритм> (дата обращения: 11.11.2021).
- 15 Язык программирования C#: краткая история, возможности и перспективы [Электронный ресурс] // Что такое C# и где он используется: обзор основных возможностей. URL: <https://timeweb.com/ru/community/articles/chto-takoe-csharp> (дата обращения: 11.11.2021).
- 16 Windows 10 [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2021. URL: https://ru.wikipedia.org/wiki/Windows_10 (дата обращения: 11.11.2021).
- 17 Microsoft Visual Studio [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2021. URL: https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio (дата обращения: 11.11.2021).
- 18 Элементы графической нотации диаграммы компонентов. [Электронный ресурс] // НОУ ИНТУИТ: [сайт]. URL: <https://intuit.ru/studies/courses/32/32/lecture/1022> (дата обращения: 19.12.2021).

ПРИЛОЖЕНИЕ А

Руководство пользователя

А.1 Назначение системы

Данная система предназначена для составления и разгадывания классического кроссворда по заданной теме. В системе предусмотрено две роли: администратор и игрок. В режиме администратора пользователю доступна возможность создания кроссворда в автоматическом или ручном режиме, а также взаимодействия со словарем понятий. В режиме игрока пользователь имеет возможность загружать кроссворд из файла, а также сохранить его в файл на любом этапе разгадывания.

А.2 Условия работы системы

Для корректной работы системы необходимо наличие соответствующих программных и аппаратных средств.

1) Требования к техническому обеспечению:

- ЭВМ типа IBM PC;
- процессор Intel Pentium не менее 1,5 ГГц (от 800 МГц);
- оперативная память 2 Гб;
- свободное место на диске 41 Мб.

2) Требования к программному обеспечению:

- операционная система Windows 7 и выше;
- установленная СУБД MS SQL Server.

А.3 Установка системы

Система поставляется в виде zip-архива. Данный файл необходимо распаковать в любую директорию на жестком диске. Запускаемым файлом системы является файл ClassicCrossword.exe.

А.4 Работа с системой

После запуска системы откроется окно авторизации (рисунок А.1), предназначенное для входа в систему. Если пользователь не зарегистрирован

в системе, то он может зарегистрироваться как новый пользователь, нажав на кнопку «Регистрация», после чего откроется окно регистрации (рисунок А.2). Для регистрации нового пользователя необходимо ввести логин, пароль и подтверждение пароля. После чего перейти обратно в окно авторизации.

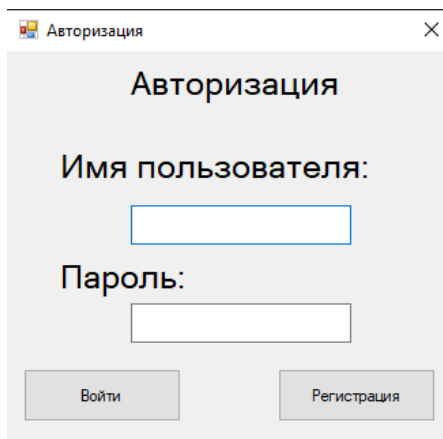


Рисунок А.1 – Экранная форма авторизации

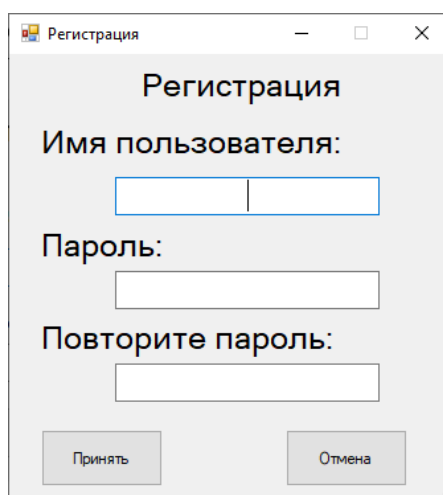


Рисунок А.2 – Экранная форма «Регистрации»

А.4.1 Работа с системой в режиме администратора

Вход в систему в роли администратора происходит с фиксированным логином «admin» и паролем «admin». Экранная форма приложения для роли администратора приведена на рисунке А.3. Администратор имеет возможность создавать кроссворд в ручном и автоматическом режиме, загружать уже существующий кроссворд для редактирования, а также сохранять его. Список понятий может быть отсортирован по алфавиту, длине, а также по маске.

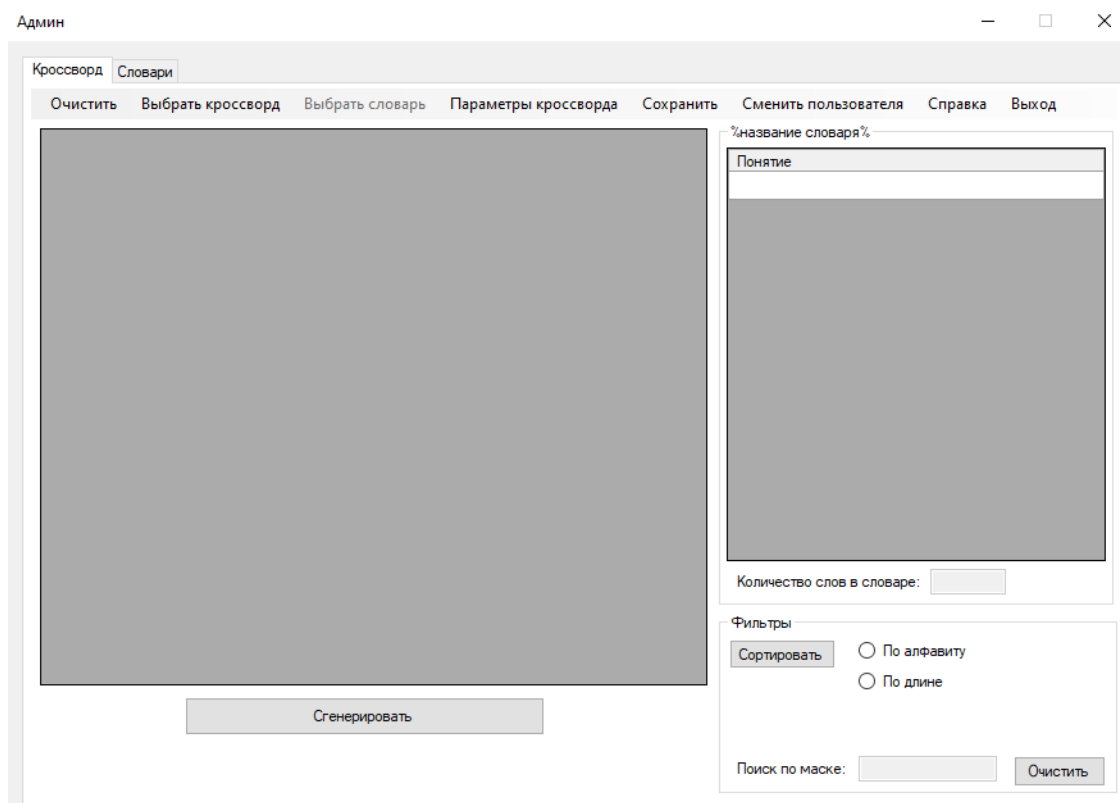


Рисунок А.3 – Главная экранная форма администратора

При нажатии на кнопку «Параметры кроссворда» администратор может задать размеры сетки по горизонтали (от 15 до 30 клеток) и вертикали (от 10 до 25 клеток), а также подключить словарь понятий, который хранится во внешнем файле формата .dict (рисунок А.4).

При нажатии на кнопку «Выбрать кроссворд» открывается диалоговое окно, где можно выбрать кроссворд (файл формата .Kros), после чего на сетке отобразится загруженный кроссворд.

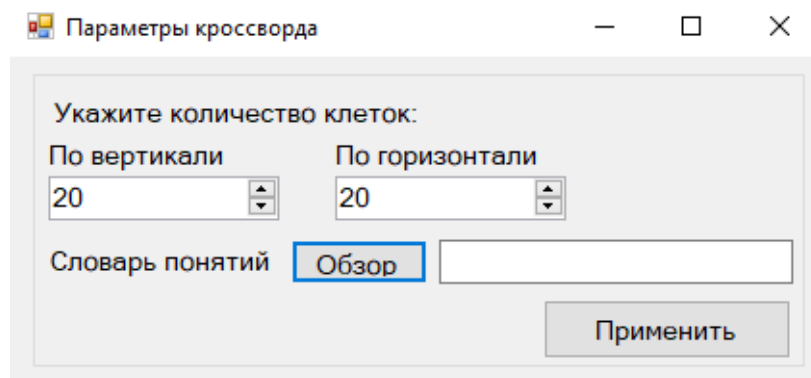


Рисунок А.4 – Экранная форма «Параметры кроссворда»

При нажатии на кнопку «Выбрать словарь» открывается диалоговое окно, где можно выбрать словарь (файл формата .dict) после чего в окне словаря понятий отобразится загруженный словарь (рисунок А.5).

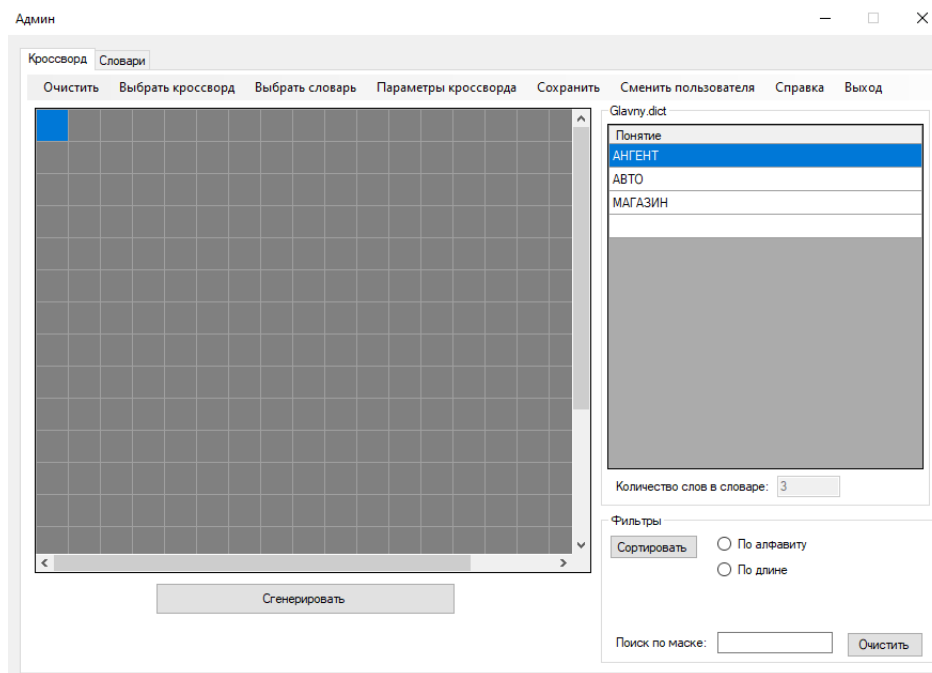


Рисунок А.5 – Экранная форма администратора с загруженным словарем понятий

Словарь понятий может быть отсортирован по алфавиту (рисунок А.6) и по длине слов (рисунок А.7).

При нажатии на кнопку «Сохранить» открывается диалоговое окно, где можно сохранить кроссворд.

При нажатии на кнопку «Создать» произойдет очистка текущей формы.

При работе с системой в автоматическом режиме достаточно задать размеры сетки в окне параметров кроссворда, указать словарь понятий, после чего нажать на кнопку «Сгенерировать», после которой выполняется генерирование кроссворда и отображение на сетке (рисунок А.8).

Glavny.dict

Понятие
АББРЕВИАТУРА
АБЕРРАЦИЯ
АБИТУРИЕНТ
АБЛЯЦИЯ
АБОЛИЦИОНИЗМ
АБОНЕМЕНТ
АБОНЕНТ
АБОРДАЖ
АБОРИГЕН
АБРАЗИВ
АБРАЗИЯ
АБСОЛЮТИЗМ
АБСОРБЦИЯ
АБСТРАКЦИОНИЗМ

Количество слов в словаре: 1000

Фильтры

Сортировать ☒ По алфавиту ☐ По длине

Поиск по маске:

Рисунок А.6 – Сортировка словаря по алфавиту

Glavny.dict

Понятие
ФОН
ФАС
ШОК
ШЕФ
ШАХ
ЭРА
БУМ
ШОУ
ФАТ
БОЙ
БАР
БОН
ФЕЯ
БРА

Количество слов в словаре: 1000

Фильтры

Сортировать ☐ По алфавиту ☒ По длине

Поиск по маске:

Рисунок А.7 – Сортировка словаря по длине

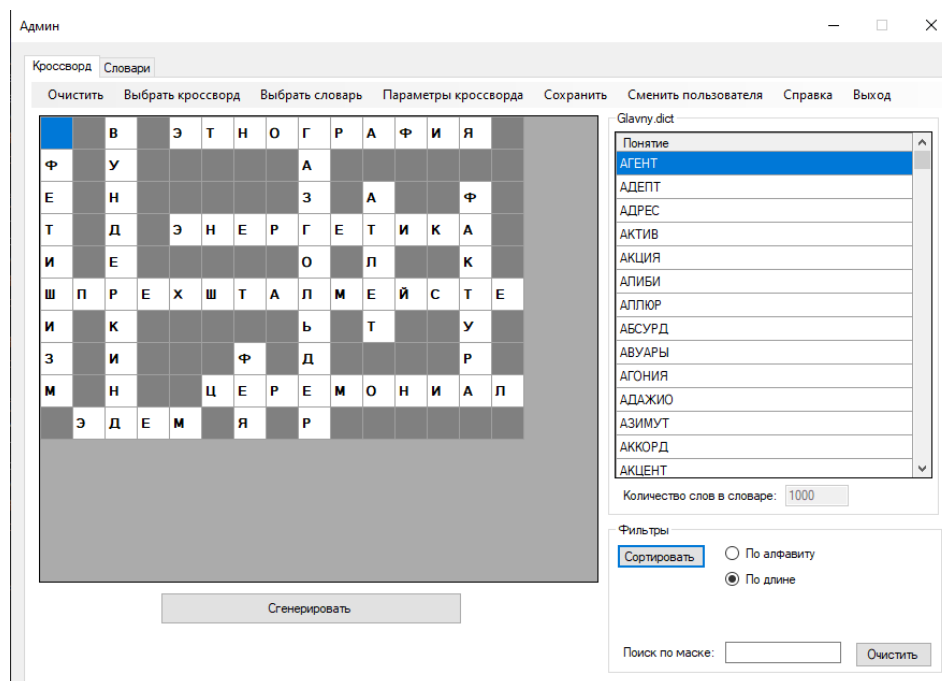


Рисунок А.8 – Пример генерации кроссворда в автоматическом режиме

При работе с системой в ручном режиме необходимо выделить на сетке рабочую область, после чего в словаре понятий произойдет отображение слов, наиболее подходящих для данной области (рисунок А.9). Для добавления слова на сетку необходимо кликнуть на него два раза.

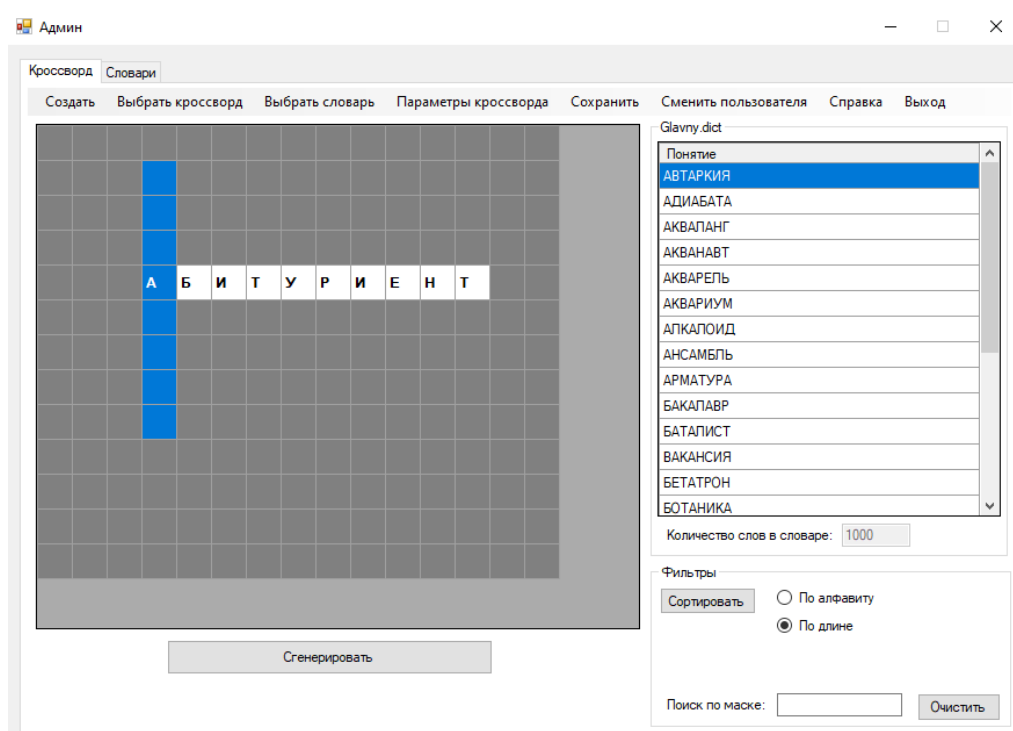


Рисунок А.9 – Пример выделения рабочей области для добавления слова

Вкладка «Словари» (рисунок А.10) предназначена для работы со словарем понятий. Здесь можно создать, загрузить и сохранить словарь. Также доступны функции добавления, изменения, удаления понятия из словаря, а также сортировки и поиска по маске.

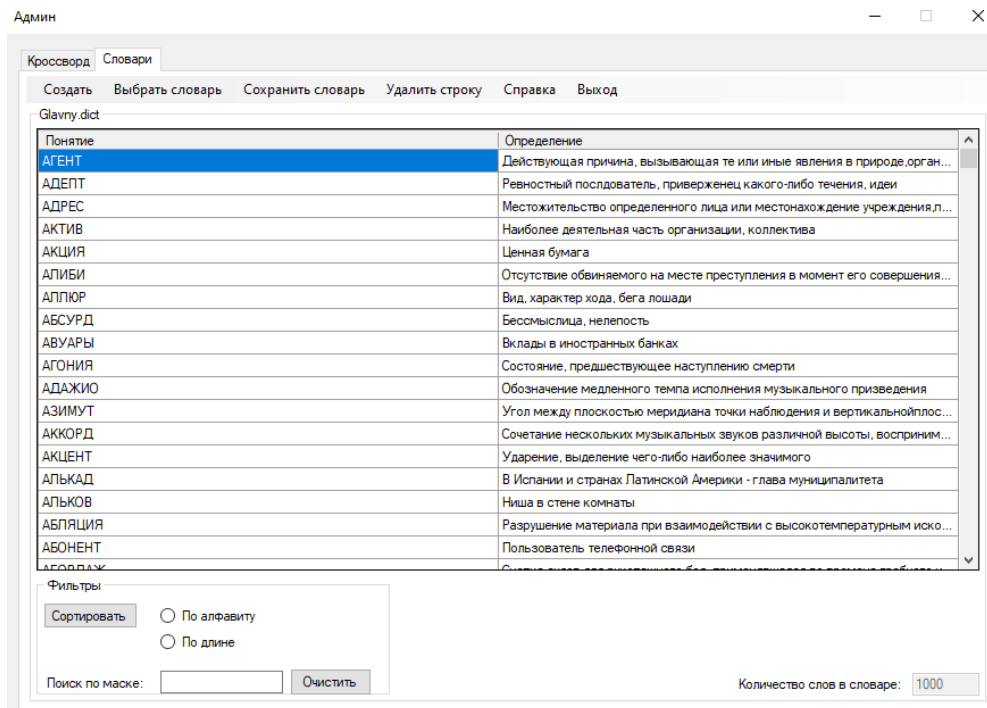


Рисунок А.10 – Экранная форма работы со словарем понятий

А.4.2 Работа с системой в режиме игрока

После авторизации пользователь переходит к форме разгадывания кроссворда, изображенной на рисунке А.11.

Пользователь должен выбрать элемент меню «Решать кроссворд» и нажать кнопку «Выбрать кроссворд» или «Загрузить сохраненный», чтобы начать/продолжить разгадывать кроссворд. Навигация по сетке кроссворда происходит с помощью щелчка мыши по выбранному квадрату сетки.

После нажатия на кнопку «Загрузить сохраненный» открывается диалоговое окно, где пользователь должен выбрать один из ранее сохраненных кроссвордов.

Пример загруженного кроссворда представлен на рисунке А.12

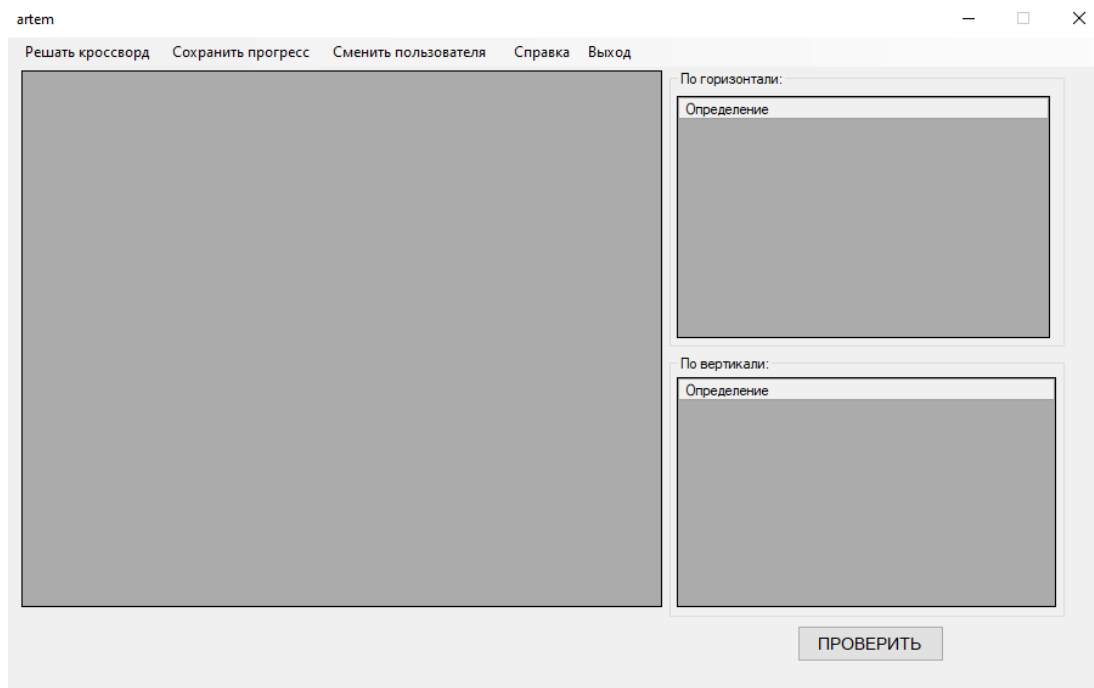


Рисунок А.11 – Экранная форма игрока

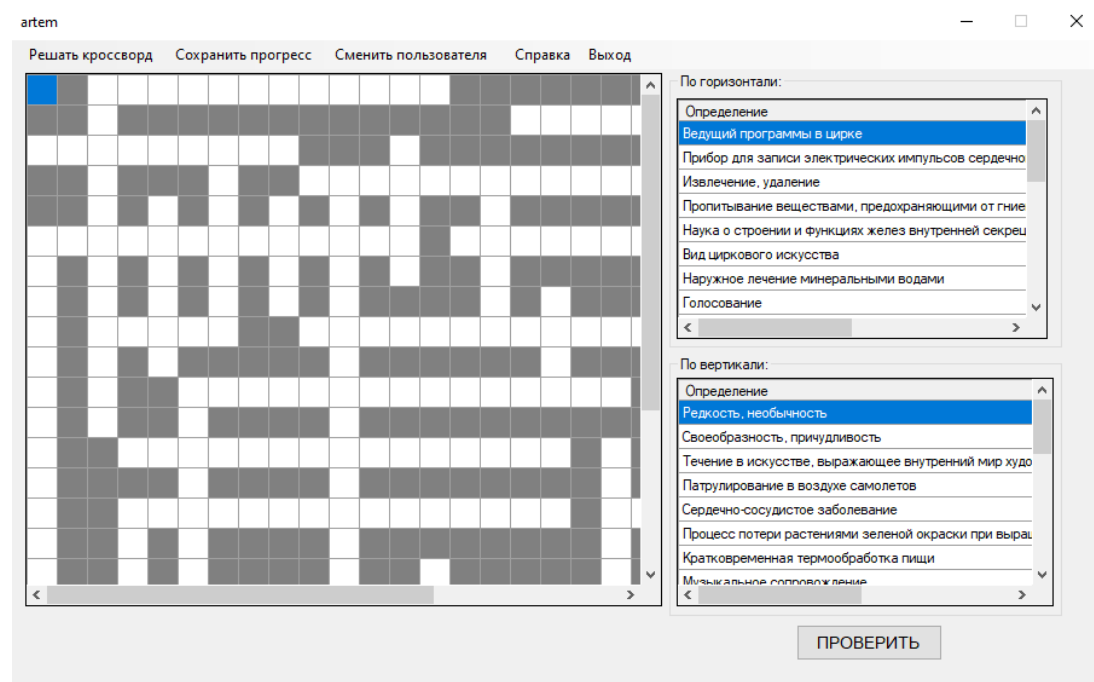


Рисунок А.12 – Пример загруженного кроссворда

При двойном щелчке на определение произойдет выделение цветом слова на сетке, после чего пользователь может производить заполнение клетки, вписывая необходимую букву (рисунок А.13). Разрешается использование только русского алфавита.

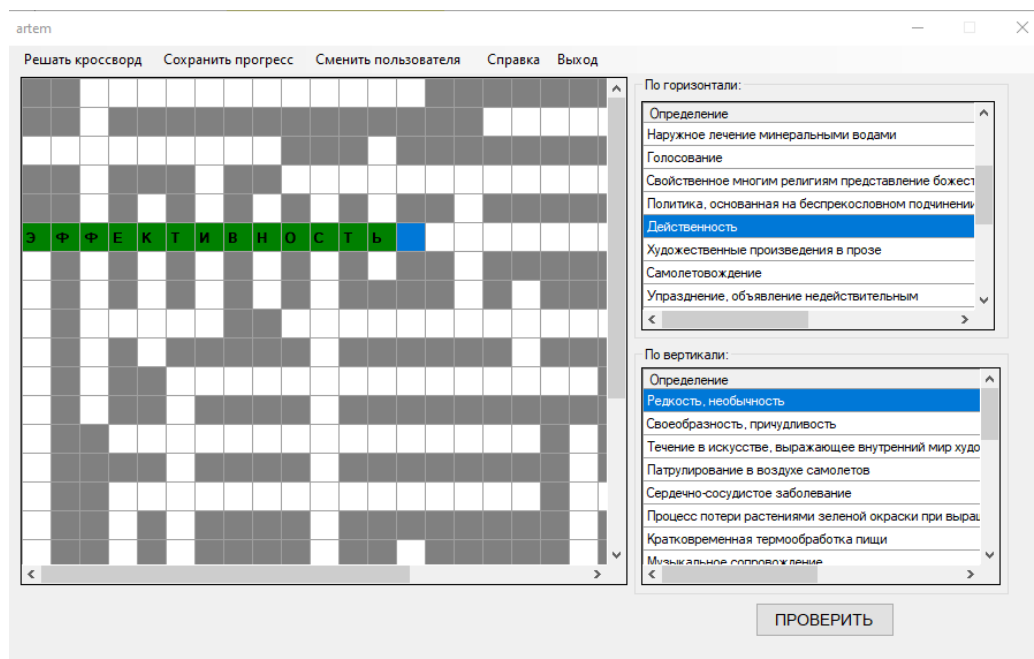


Рисунок А.13 – Пример заполнения кроссворда

Также пользователь имеет возможность сохранить кроссворд на любом этапе разгадывания, для этого пользователь должен нажать на кнопку «Сохранить прогресс».

После решения кроссворда, пользователь получит уведомление о правильности или неправильности решения кроссворда (рисунок А.14).

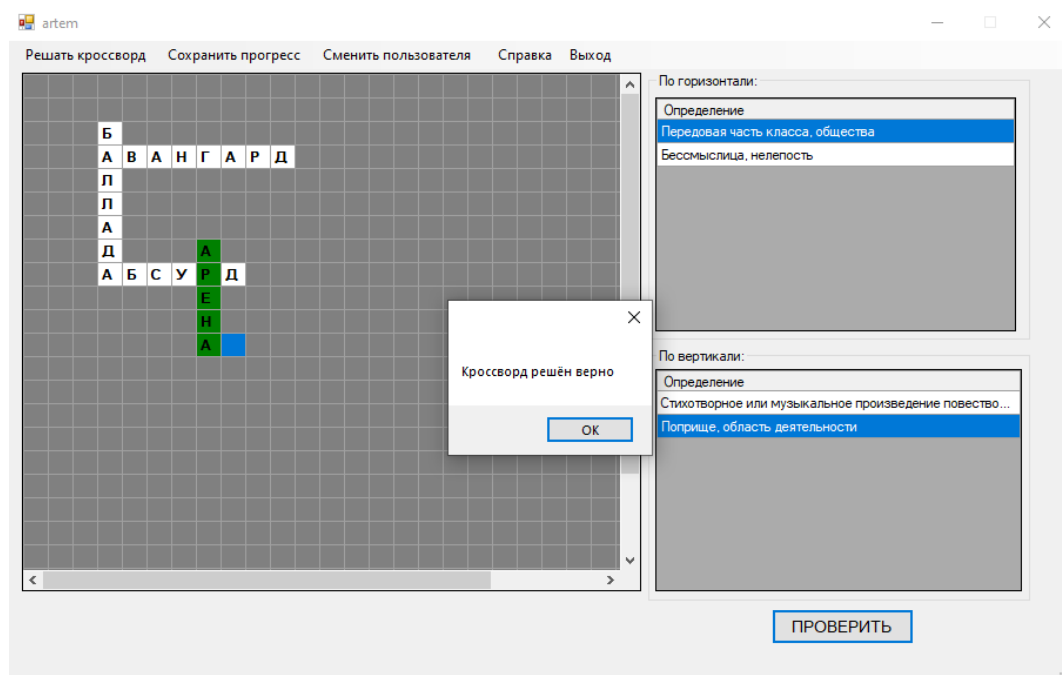


Рисунок А.14 – Решенный кроссворд

ПРИЛОЖЕНИЕ Б

Листинг модулей программы

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;

namespace ClassicCrossword.Controller
{
    /// <summary>
    /// Класс, для работы со словарем понятий
    /// </summary>
    public class DictController
    {
        /// <summary>
        /// Метод чтения словаря понятий
        /// </summary>
        /// <param name="filename">Имя файла</param>
        /// <returns></returns>
        public Dictionary<string, string> ParseDict(String filename)
        {
            Dictionary<string, string> dict = new Dictionary<string, string>();
            string[] words = File.ReadAllLines(filename, Encoding.GetEncoding("utf-
8")).Take(1000).ToArray();
            for (int i = 0; i < words.Length; i++)
            {
                string word = words[i].Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries)[0];
                string question = words[i].Substring(words[i].IndexOf(words[i].Split(new char[] { ' ' },
StringSplitOptions.RemoveEmptyEntries)[1]));
                var isHere = dict.TryGetValue(word, out var alreadyHereworld);
                if (isHere)
                { }
                else
                {
                    dict.Add(word, question);
                }
            }
            return dict;
        }

        /// <summary>
        /// Метод, сохраняющий словарь понятий
        /// </summary>
        /// <param name="text">Содержимое словаря</param>
        /// <param name="filename">Имя файла</param>
        /// <returns></returns>
        public bool SaveDict(String text, String filename)
        {
            try
            {
                using (StreamWriter sw = new StreamWriter(filename, false, System.Text.Encoding.UTF8))
                {
                    sw.Write(text);
                    return true;
                }
            }
            catch { return false; }
        }
    }
}
```

```

using ClassicCrossword.Forms;
using ClassicCrossword.Model;
using System;
using System.Data;
using System.Data.SqlClient;

namespace ClassicCrossword.DAO
{
    /// <summary>
    /// Класс для добавления пользователей в базу данных
    /// </summary>
    public class UserDAO
    {
        /// <summary>
        /// Метод добавления игрока
        /// </summary>
        /// <param name="player">Игрок</param>
        /// <returns></returns>
        public bool Insert(Player player)
        {
            try
            {
                if (!HasSameType(player, false))
                {
                    SqlConnection sqlConnection = ConnectionDB.Connect();
                    string sql = string.Format("Insert into Player (login, pass) Values (@player_login,
@player_pass);");

                    using (SqlCommand cmd = new SqlCommand(sql, sqlConnection))
                    {
                        SqlParameter param = new SqlParameter();
                        param.ParameterName = "@player_login";
                        param.Value = player.Login;
                        param.SqlDbType = SqlDbType.VarChar;
                        param.Size = 100;
                        cmd.Parameters.Add(param);

                        param = new SqlParameter();
                        param.ParameterName = "@player_pass";
                        param.Value = player.Pass;
                        param.SqlDbType = SqlDbType.VarChar;
                        param.Size = 100;
                        cmd.Parameters.Add(param);

                        cmd.ExecuteNonQuery();
                    }
                    ConnectionDB.Disconnect(sqlConnection);
                    return true;
                }
                else return false;
            }
            catch (SqlException ex)
            {
                throw ex;
            }
        }

        /// <summary>
        /// Метод получения зарегистрированного пользователя
        /// </summary>
        /// <param name="login">Логин</param>
        /// <param name="password">Пароль</param>
        /// <returns></returns>
    }
}

```

```

public User GetUserByAuthorization(string login, string password)
{
    User user = null;
    try
    {
        SqlConnection sqlConnection = ConnectionDB.Connect();
        string sql = string.Format("Select login, pass From Player Where login= Lower('{0}') AND
pass='{1}'", login.ToLower(), password);

        SqlCommand cmd = sqlConnection.CreateCommand();
        cmd.CommandText = sql;
        SqlDataReader dataReader = cmd.ExecuteReader();
        while (dataReader.Read())
        {
            user = new Player(dataReader[0].ToString(), dataReader[1].ToString());
        }
        dataReader.Close();
        ConnectionDB.Disconnect(sqlConnection);
    }
    catch (SqlException ex)
    {
        throw ex;
    }
    return user;
}

private bool HasSameType(Player player, bool isUpdate)
{
    try
    {
        SqlConnection sqlConnection = ConnectionDB.Connect();
        string sql = string.Format("Select count(id) From Player Where UPPER(REPLACE(login,'
;'))=UPPER(REPLACE('{0}',';'))", player.Login);
        if (isUpdate)
            sql = string.Format("Select count(id) From Player Where UPPER(REPLACE(login,'
;'))=UPPER(REPLACE('{0}',';')) AND id!='{1}'", player.Login, player.Id);
        SqlCommand cmd = sqlConnection.CreateCommand();
        cmd.CommandText = sql;
        SqlDataReader dataReader = cmd.ExecuteReader();
        int count = -1;
        while (dataReader.Read())
        {
            count = Convert.ToInt32(dataReader[0]);
        }
        if (count == 1) RegistrationForm.flag = false;
        dataReader.Close();
        ConnectionDB.Disconnect(sqlConnection);
        if (count > 0) return true;
        else
            return false;
    }
    catch (SqlException ex)
    {
        throw ex;
    }
}

}

using System;
using System.Configuration;

```

```

using System.Data.SqlClient;

namespace ClassicCrossword.DAO
{
    /// <summary>
    /// Класс для подключения к базе данных
    /// </summary>
    static class ConnectionDB
    {
        /// <summary>
        /// Метод подключения к БД
        /// </summary>
        /// <returns></returns>
        public static SqlConnection Connect()
        {
            SqlConnection sqlConnection = null;
            try
            {
                string connectionString =
ConfigurationManager.ConnectionStrings["CrosswordDBConnectionString"].ConnectionString;
                sqlConnection = new SqlConnection(connectionString);
                sqlConnection.Open();
            }
            catch (Exception ex)
            {
                throw ex;
            }
            return sqlConnection;
        }

        /// <summary>
        /// Метод закрытия соединения
        /// </summary>
        /// <param name="sqlConnection"></param>
        public static void Disconnect(SqlConnection sqlConnection)
        {
            try
            {
                if (sqlConnection != null)
                {
                    sqlConnection.Close();
                }
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }
    }
}

using System;
using System.Collections.Generic;

namespace ClassicCrossword.Model
{
    /// <summary>
    /// Класс работы с кроссвордом
    /// </summary>
    [Serializable]
    public class Crossword
    {
        const string Letters = "абвгдеёжзийклмнопрстуфхцчщъыьэюя";
    }
}

```

```

readonly int[,] _hWords;
readonly int[,] _vWords;
readonly int[] _dirX = { 0, 1 };
readonly int[] _dirY = { 1, 0 };
private char[,] _board;
int _hCount, _vCount;
private static IList<string> _wordsToInsert;
private static char[,] _tempBoard;
Random _rand = new Random();
private List<string> temp = new List<string>();

public char this[int i, int j]
{
    get { return _board[i, j]; }
    set { _board[i, j] = value; }
}

public int N { get; set; }

public int M { get; set; }

public char[,] GetBoard
{
    get { return _board; }
    set { _board = value; }
}

public List<string> Temp
{
    get { return temp; }
    set { temp = value; }
}

public Dictionary<string, string> defHorDict = new Dictionary<string, string>();
public Dictionary<string, string> defVerDict = new Dictionary<string, string>();

public char[,] progressPlayer { get; set; }

public Crossword() {}

public Crossword(int xDimen, int yDimen)
{
    _board = new char[xDimen, yDimen];
    _hWords = new int[xDimen, yDimen];
    _vWords = new int[xDimen, yDimen];
    N = xDimen;
    M = yDimen;
    _rand = new Random();

    for (var i = 0; i < N; i++)
        for (var j = 0; j < M; j++)
            _board[i, j] = ' ';
}

//Проверка позиции
bool IsValidPosition(int x, int y)
{
    return x >= 0 && y >= 0 && x < N && y < M;
}

int CanBePlaced(string word, int x, int y, int dir)
{

```

```

var result = 0;
if (dir == 0)
{
    for (var j = 0; j < word.Length; j++)
    {
        int x1 = x, y1 = y + j;
        if (!IsValidPosition(x1, y1) && (_board[x1, y1] == '' || _board[x1, y1] == word[j])) //выход
за границы или позиция занята, тогда -1
            return -1;
        if (IsValidPosition(x1 - 1, y1))
            if (_hWords[x1 - 1, y1] > 0) //рядом позиция с буквой => вставлять нельзя => -1
                return -1;
        if (IsValidPosition(x1 + 1, y1))
            if (_hWords[x1 + 1, y1] > 0) //рядом позиция с буквой => вставлять нельзя => -1
                return -1;
        if (_board[x1, y1] == word[j]) //считаем кол-во пересечений
            result++;
    }
}

else
{
    for (var j = 0; j < word.Length; j++)
    {
        int x1 = x + j, y1 = y;
        if (!IsValidPosition(x1, y1) && (_board[x1, y1] == '' || _board[x1, y1] == word[j])) //выход
за границы или позиция занята, тогда -1
            return -1;
        if (IsValidPosition(x1, y1 - 1))
            if (_vWords[x1, y1 - 1] > 0) //рядом позиция с буквой => вставлять нельзя => -1
                return -1;
        if (IsValidPosition(x1, y1 + 1))
            if (_vWords[x1, y1 + 1] > 0) //рядом позиция с буквой => вставлять нельзя => -1
                return -1;
        if (_board[x1, y1] == word[j]) //считаем кол-во пересечений
            result++;
    }
}

int xStar = x - _dirX[dir], yStar = y - _dirY[dir];
if (IsValidPosition(xStar, yStar))
    if (!(_board[xStar, yStar] == '' || _board[xStar, yStar] == '*'))
        return -1;

xStar = x + _dirX[dir] * word.Length;
yStar = y + _dirY[dir] * word.Length;
if (IsValidPosition(xStar, yStar))
    if (!(_board[xStar, yStar] == '' || _board[xStar, yStar] == '*')) //позиция занята
        return -1;

return result == word.Length ? -1 : result;
}

void PutWord(string word, int x, int y, int dir, int value)
{
    var mat = dir == 0 ? _hWords : _vWords;

    for (var i = 0; i < word.Length; i++)
    {
        int x1 = x + _dirX[dir] * i, y1 = y + _dirY[dir] * i;
        _board[x1, y1] = word[i];
        mat[x1, y1] = value;
    }
}

```



```

    }

    int xStar = x - _dirX[dir], yStar = y - _dirY[dir];
    if (IsValidPosition(xStar, yStar)) _board[xStar, yStar] = '*';
    xStar = x + _dirX[dir] * word.Length;
    yStar = y + _dirY[dir] * word.Length;
    if (IsValidPosition(xStar, yStar)) _board[xStar, yStar] = '*';
}

//Добавление слова
public int AddWord(string word, string def)
{
    var wordToInsert = word;
    var info = BestPosition(wordToInsert);
    if (info != null)
    {
        if (info.Item3 == 0)
            _hCount++;
        else
            _vCount++;
        var value = info.Item3 == 0 ? _hCount : _vCount;
        PutWord(wordToInsert, info.Item1, info.Item2, info.Item3, value);

        if (info.Item3 == 0)
        {
            defHorDict.Add(word, def);
        }
        else if (info.Item3 == 1)
        {
            defVerDict.Add(word, def);
        }

        return info.Item3;
    }
    return -1;
}

//Добавление слова
public int AddWord(string word, string def, int x, int y, int dir)
{
    if (dir == 0)
        _hCount++;
    else
        _vCount++;
    var value = dir == 0 ? _hCount : _vCount;
    PutWord(word, x, y, dir, value);

    if (dir == 0)
    {
        defHorDict.Add(word, def);
    }
    else if (dir == 1)
    {
        defVerDict.Add(word, def);
    }

    return dir;
}

//Поиск позиции
List<Tuple<int, int, int>> FindPositions(string word)
{
    var max = 0;

```

```

var positions = new List<Tuple<int, int, int>>();
for (var x = 0; x < N; x++)
{
    for (var y = 0; y < M; y++)
    {
        for (var i = 0; i < _dirX.Length; i++)
        {
            var dir = i;
            var wordToInsert = word;
            var count = CanBePlaced(wordToInsert, x, y, dir);

            if (count < max)
                continue; //если кол-во позиций <, чем уже было, то идем на сл итерацию
            if (count > max) //очищаем весь список позиций, если нашли позиции для
                вставки, где их больше всего
                positions.Clear();
            max = count;
            positions.Add(new Tuple<int, int, int>(x, y, dir));
        }
    }
}
if (max == 0 && !Temp[0].Equals(word))
    positions.Clear();
return positions;
}

Tuple<int, int, int> BestPosition(string word)
{
    var positions = FindPositions(word);
    if (positions.Count > 0)
    {
        var index = _rand.Next(positions.Count);
        return positions[index];
    }
    return null;
}

public bool IsLetter(char a)
{
    return Letters.Contains(a.ToString());
}

//очистка поля
public void Reset()
{
    for (var i = 0; i < N; i++)
    {
        for (var j = 0; j < M; j++)
        {
            _board[i, j] = ' ';
            _vWords[i, j] = 0;
            _hWords[i, j] = 0;
            _hCount = _vCount = 0;
        }
    }
    defHorDict.Clear();
    defVerDict.Clear();
}

private void RemoveWord(string word, int x, int y, int dir)
{
    var mat = dir == 0 ? _hWords : _vWords;
    var mat1 = dir == 0 ? _vWords : _hWords;

```

```

for (var i = 0; i < word.Length; i++)
{
    int x1 = x + _dirX[dir] * i, y1 = y + _dirY[dir] * i;
    if (mat1[x1, y1] == 0)
        _board[x1, y1] = ' ';
    mat[x1, y1] = 0;
}

int xStar = x - _dirX[dir], yStar = y - _dirY[dir];
if (IsValidPosition(xStar, yStar) && HasFactibleValueAround(xStar, yStar))
    _board[xStar, yStar] = ' ';

xStar = x + _dirX[dir] * word.Length;
yStar = y + _dirY[dir] * word.Length;
if (IsValidPosition(xStar, yStar) && HasFactibleValueAround(xStar, yStar))
    _board[xStar, yStar] = ' ';
}

bool HasFactibleValueAround(int x, int y)
{
    for (var i = 0; i < _dirX.Length; i++)
    {
        int x1 = x + _dirX[i], y1 = y + _dirY[i];
        if (IsValidPosition(x1, y1) && (_board[x1, y1] != ' ' || _board[x1, y1] == '*'))
            return true;
        x1 = x - _dirX[i];
        y1 = y - _dirY[i];
        if (IsValidPosition(x1, y1) && (_board[x1, y1] != ' ' || _board[x1, y1] == '*'))
            return true;
    }
    return false;
}
}
}

```