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, математики и электроники

Факультет информатики

Кафедра программных систем

**ОТЧЕТ**  
  
 к лабораторному практикуму

по дисциплине «Технологии программирования»

по теме «Автоматизированная система составления и

разгадывания кроссворда»

Студент В.А. Байрамов

Студент В.Д. Мавлютов

Студент Н.Д. Перевозчиков

Студент С.В. Федоров

Руководитель Л.С. Зеленко

Самара 2018

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, математики и электроники

Факультет информатики

Кафедра программных систем

**ЗАДАНИЕ**

на лабораторный практикум по дисциплине

«Технологии программирования»

студентам группы № 6402-090301D

В.А. Байрамов  
В.Д. Мавлютов   
Н.Д. Перевозчиков  
С.В. Федоров

1. **Тема проекта:** «Автоматизированная система составления и разгадывания линейного кроссворда по выбранной теме»
2. **Исходные данные к проекту:** см. приложение к заданию
3. **Перечень вопросов, подлежащих разработке:**
   1. Произвести анализ предметной области: изучить основные принципы составления кроссвордов, изучить алгоритмы генерации кроссвордов
   2. Выполнить обзор существующих систем-аналогов
   3. Разработать информационно-логический проект системы
   4. Разработать и реализовать программное и информационное обеспечение, провести его тестирование и отладку .
   5. Оформить документацию проекта
   6. Подготовить презентацию по разработанной системе
4. **Перечень графических разработок**
   1. Структурная схема системы
   2. Диаграмма классов, диаграмма модулей (компонентов)
   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 | 09.10.2018 |  |
| 2 | Описание и анализ предметной области (1 раздел) | 10 | 16.10.2018 |  |
| 3 | Проектирование системы (2 раздел) | 30 | 04.12.2018 |  |
| 3.1 | Разработка структурной схемы системы | 5 | 16.10.2018 |  |
| 3.2 | Разработка функциональной спецификации системы | 10 | 06.11.2018 |  |
| 3.3 | Разработка прототипов экранных форм | 10 | 06.11.2018 |  |
| 3.4 | Разработка основных алгоритмов | 5 | 04.12.2018 |  |
| 4 | Реализация проекта, разработка контрольных примеров. Предъявление реализации руководителю (3 раздел) | 45 | 11.12.2018 |  |
| 5 | Корректировка проекта и оформление документации проекта. Защита проекта с представлением презентации | 10 | 25.12.2018 |  |

Задание принял  
 к исполнению В.А. Байрамов

В.Д. Мавлютов

Н.Д. Перевозчиков

С.В. Федоров

**ПРИЛОЖЕНИЕ  
к заданию на лабораторный практикум**студентам группы № 6402-090301D  
В.А. Байрамов

В.Д. Мавлютов

Н.Д. Перевозчиков

С.В. Федоров

Тема проекта: «**Автоматизированная система составления и разгадывания классического кроссворда»**

**Исходные данные к проекту:**

1. **Характеристика объекта автоматизации:**

## объект автоматизации: классический кроссворд;

## виды автоматизируемой деятельности:

* + процесс создания/генерирования кроссворда;
  + процесс разгадывания кроссворда;
  + процесс работы со словарем понятий;
  + процесс визуализации работы с кроссвордом;

## количество ролей пользователей – 2;

## минимальная ширина кроссворда – 7 символов;

## максимальная ширина кроссворда – 20 символов;

## минимальная высота кроссворда – 7 символов;

## максимальная высота кроссворда – 20 символов;

1. минимальная длина слова – 3 символа;
2. максимальная длина слова – 20 символов;

## количество режимов создания – 2;

1. минимальное количество букв в пересечении – 1;
2. максимальное количество букв в пересечении – 9;

## количество видов сортировки словаря понятий – 2;

## минимальное количество подсказок – 1;

## максимальное количество подсказок – 10% от количества слов;

## язык записи понятий – русский.

1. **Требования к информационному обеспечению:**
2. информационное обеспечение разрабатывается на основе следующего источника:
   * Описание структуры кроссворда [Электронный ресурс]. URL: ru.wikipedia.org/wiki/Кроссворд (дата обращения: 01.10.2018);
3. структура словаря понятий (понятие и его определение располагаются в одной строке, разделены пробелом);
4. словари понятий хранятся в текстовых файлах формата \*.dict;
5. кроссворды хранятся в файлах, структура файла определяется в процессе проектирования.
6. **Требования к техническому обеспечению:**
7. тип ЭВМ – IBM PC совместимый;
8. монитор с разрешающей способностью не ниже 800 х 600;
9. манипулятор – мышь;
10. технические характеристики определяются в процессе выполнения проекта.
11. **Требования к программному обеспечению:**
12. тип операционной системы – Windows 7/8 и выше;
13. язык программирования – С#;
14. среда программирования – Visual Studio 2017;
15. среда проектирования – StarUML 2.8.0.
16. **Общие требования к проектируемой системе:**

***5.1 Функции, реализуемые системой:***

1. *общесистемные функции:*
   * авторизация 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;
   * контроль количества взятых подсказок;
   * проверка правильности разгадывания кроссворда;
   * выдача справочной информации о системе;
2. *функции администратора:*
   * настройка параметров кроссворда при создании:
3. задание высоты;
4. задание ширины;
5. подключение словаря понятий;
   * составление/редактирование кроссворда:
6. добавление слова;
7. удаление слова;
8. изменение ориентации слова;
9. фильтрация словаря понятий по маске;
10. сортировка словаря;
    * сохранение кроссворда в файл заданной структуры;
    * загрузка кроссворда из файла;
    * работа со словарями понятий:
11. добавление понятия;
12. удаление понятия;
13. изменение понятия;
14. проверка дублирования понятий;
15. проверка языка записи понятий;
16. сортировка словаря по выбранному критерию;
17. поиск по заданной маске;
18. загрузка словаря из файла;
19. сохранение словаря в файл;
20. создание нового словаря понятий;
21. *функции пользователя:*
    * загрузка кроссворда из файла;
    * разгадывание кроссворда с организацией системы подсказок:
22. выбор слова;
23. вписывание/удаление/изменение буквы;
24. взятие подсказки;
    * сохранение кроссворда в файл.

***5.2 Технические требования к системе:***

1. режим работы – диалоговый;
2. время автоматической генерации кроссворда – не более 50 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°С;
6. влажность воздуха – 45-75%.

Руководитель   
проекта Л.С. Зеленко

Задание принял  
к исполнению В.А. Байрамов

В.Д. Мавлютов

Н.Д. Перевозчиков

С.В. Федоров

**РЕФЕРАТ**

Пояснительная записка 85 с, 50 рисунков, 5 таблиц[[1]](#footnote-1), 30 источников,  
2 приложения.

Графическая часть: 20 слайдов презентации PowerPoint.

КРОССВОРД КЛАССИЧЕСКИЙ, ГЕНЕРАТОР КРОССВОРДОВ, ГОЛОВОЛОМКА, СЛОВАРЬ ПОНЯТИЙ, РАЗГАДЫВАНИЕ, ВИЗУАЛИЗАЦИЯ

Во время лабораторного практикума разработаны алгоритмы и соответствующая им программа, позволяющая выполнять автоматическую генерацию классического кроссворда. 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 С# в среде Visual Studio 2017 и функционирует под управлением операционной системы Windows 7/8/10.

.

**СОДЕРЖАНИЕ**

[1 Описание и анализ предметной области 13](#_Toc532998287)

[1.1 Классификация кроссвордов 13](#_Toc532998288)

[1.2 Правила построения кроссвордов 15](#_Toc532998289)

[1.3 Описание систем-аналогов 16](#_Toc532998290)

[1.3.1 Приложение «Сrosswordus» 16](#_Toc532998291)

[1.3.2 Приложение «Puzzlecup» 17](#_Toc532998292)

[1.3.3 Приложение «Hot Potatoes» 18](#_Toc532998293)

[1.4 Диаграмма объектов предметной области 19](#_Toc532998294)

[1.5 Постановка задачи 20](#_Toc532998295)

[2 Проектирование системы 24](#_Toc532998296)

[2.1 Структурная схема системы 24](#_Toc532998297)

[2.2 Спецификация системы 26](#_Toc532998298)

[2.2.1 Функциональная спецификация 27](#_Toc532998299)

[2.2.2 Спецификация качества 27](#_Toc532998300)

[2.2.3 Перечень исключительных ситуаций 36](#_Toc532998301)

[2.2.4 Разработка прототипа интерфейса пользователя системы 37](#_Toc532998302)

[2.3 Разработка структур данных и классов 46](#_Toc532998303)

[2.4 Разработка и описание алгоритмов обработки данных 48](#_Toc532998304)

[2.5 Выбор и обоснование комплекса программных средств 51](#_Toc532998305)

[2.5.1 Выбор языка программирования и среды разработки 51](#_Toc532998306)

[2.5.2 Выбор операционной системы 52](#_Toc532998307)

[2.5.3 Выбор среды программирования 52](#_Toc532998308)

[3 Реализация системы 53](#_Toc532998309)

[3.1 Разработка и описание интерфейса пользователя 53](#_Toc532998310)

[3.1.1 Режим пользователя 53](#_Toc532998311)

[3.1.2 Режим администратора 54](#_Toc532998312)

[3.1.3 Описание тестового примера 58](#_Toc532998313)

[3.2 Реализация классов и структур данных 61](#_Toc532998314)

[3.3 Реализация и описание модулей программы 61](#_Toc532998315)

[3.4 Выбор и обоснование комплекса технических средств 61](#_Toc532998316)

[3.4.1 Расчет объема занимаемой памяти 61](#_Toc532998317)

[Расчет объема внешней памяти 61](#_Toc532998318)

[Расчет объема ОЗУ 62](#_Toc532998319)

[3.4.2 Минимальные требования, предъявляемые к системе 63](#_Toc532998320)

[ПРИЛОЖЕНИЕ А Руководство пользователя 68](#_Toc532998321)

[А.1 Назначение системы 68](#_Toc532998322)

[А.2 Условия работы системы 68](#_Toc532998323)

[А.3 Установка системы 68](#_Toc532998324)

[А.4 Работа с системой 68](#_Toc532998325)

[А.4.1 Работа с системой в режиме администратора (если необходимо) 69](#_Toc532998326)

[Вход в систему (авторизация) 69](#_Toc532998327)

[А.4.2 Работа с системой в режиме пользователя 69](#_Toc532998328)

[Вход в систему (авторизация) 69](#_Toc532998329)

[Вход в систему (регистрация) 69](#_Toc532998330)

[Настройка параметров кроссворда 69](#_Toc532998331)

[ПРИЛОЖЕНИЕ Б Листинг модулей программы 70](#_Toc532998332)

ВВЕДЕНИЕ

Кроссворд – головоломка, представляющая собой переплетение рядов клеточек, которые заполняются словами по заданным значениям. Обычно значения слов задаются описательно под этой схемой, сначала значения слов, которые должны получиться по горизонтали, затем – по вертикали [1].

Исследователями встречались находки, похожие на кроссворд, датированные еще 1-4 вв. н. э. По одной из версий, прототипы современных кроссвордов появились еще в XIX веке. Самый первый дошедший до нас кроссворд был опубликован в 1875 году в номере журнала «Святой Николас» в Нью-Йорке.

Кроссворд продолжает 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.

В общем случае кроссворды, как и другие интеллектуальные головоломки, – это полезное увлечение. Они расширяют кругозор, тренируют память и учат выуживать из нее необходимые сведения.

Для того чтобы повысить эрудицию в конкретных областях деятельности, можно выбирать тематические кроссворды. Подобные головоломки также используются в обучении, например, иностранным языкам – они помогают запоминать новые слова.

Перед авторами поставлена задача – разработать автоматизированную систему «Классический кроссворд» с возможностью составления кроссворда в автоматическом и ручном режимах и его разгадывания.

Разработка системы будет вестись по технологии RAD (Rapid Application Development), которая поддерживается методологией объектно-ориентированной декомпозиции предметной области и принципами структурного проектирования. Модель RAD очень хорошо подходит к разработке учебных программ, т.к. включает в себя три составляющие [2]:

1. Небольшую команду программистов (от 2 до 4 человек);
2. Короткий, но тщательно проработанный производственный график (от 2 до 4 мес.);
3. Повторяющийся цикл, при котором разработчики по мере того, как приложение начинает обретать форму, запрашивают и реализуют в продукте требования, полученные через взаимодействие с заказчиком.

При проектировании системы буде использоваться методология ООАП (Объектно-ориентированный анализ и проектирование). Объектно-ориентированный анализ – это методология анализа предметной области, при которой требования к проектируемой системе воспринимаются с точки зрения классов и объектов, выявленных в предметной области [3].

Объектно-ориентированное проектирование – это методология проектирования, соединяющая в себе процесс объектной декомпозиции и приемы представления логической и физической, а также статической и динамической моделей проектируемой системы.

* + 1. Описание и анализ предметной области

Предметная область – часть реального мира, которая имеет существенное значение или непосредственное отношение к процессу функционирования программы [4]. Предметная область включает в себя только те объекты и взаимосвязи между ними, которые необходимы для описания требований и условий решения некоторой задачи.

* 1. Классификация кроссвордов

Кроссворд – игра-задача, в которой фигура из рядов пустых клеток заполняется перекрещивающимися словами со значениями, заданными по условию игры [5].

Существует большое количество видов кроссвордов. Самые распространенные из них [6]:

* Классические кроссворды. В отличие от сканвордов, в классических кроссвордах все вопросы представлены в виде отдельного пронумерованного списка, разделенного на две части: «по горизонтали», по «вертикали». Ответы необходимо вносить в поле кроссворда в соответствующие блоки из пустых ячеек слева направо и сверху вниз, как правило, в именительном падеже в единственном числе, за исключением слов, правописание которых предусматривает только множественное число. На рисунке 1 приведен классический кроссворд.

  
Рисунок 1 – Пример классического кроссворда

* Сканворды. В этих кроссвордах все вопросы внесены прямо в сетку самого кроссворда. При помощи стрелок-указателей определяется расположение разгадываемых слов. В отличие от классических кроссвордов, разгадываемые слова в сканвордах имеют большее количество взаимных пересечений, что в некоторой степени упрощает их разгадывание. В связи с ограниченным пространством для написания вопросов, в сканвордах очень распространены вопросы-задания, построенные на различных ассоциациях, словах-антагонистах, словах-синонимах, определениях пропущенных слов в различных известных словосочетаниях. Пример сканворда представлен на рисунке 2.



Рисунок 2 – Пример сканворда

* Венгерские кроссворды (филворды). Филворд представляет собой поле, заполненное буквами. Из всей этой комбинации букв необходимо отыскать слова, которые указываются в виде отдельного списка или отыскать слова-ответы на ряд вопросов, приложенных к кроссворду. Искомые слова в поле филворда могут располагаться в любом направлении в виде ломаной под прямыми углами линии. Каждая из букв может быть использована только один раз. Пример венгерского кроссворда представлен на рисунке 3.



Рисунок 3 – Пример венгерского кроссворда

* 1. Правила построения кроссвордов

Кроссворд не имеет строгих правил и жестких ограничений, но есть традиции, которых придерживается большинство «кроссвордных» изданий. Обычно, когда упоминаются «правила кроссворда», имеется в виду именно этот негласный стандарт, и уточняются отклонения [7].

Суть кроссворда состоит в разгадывании слов по определениям. К каждому слову дается текстовое определение, в описательной или вопросительной форме указывающее некое слово, являющееся ответом. Ответ вписывается в сетку и, благодаря пересечениям с другими словами, облегчает нахождение ответов на другие определения.

Загаданные слова представляются в виде цепочки ячеек, в каждую из которых по порядку вписываются буквы ответа – по одной в каждую ячейку. Слова пересекаются друг с другом, образуя сетку. Классическая сетка состоит из слов, написанных по вертикали и горизонтали. Сетка не должна иметь изолированных участков.

Слова-ответы должны быть существительными в именительном падеже и единственном числе. Множественное число допускается, когда оно обозначает единственный предмет.

Хорошим тоном считается симметрия сетки кроссворда относительно вертикальной, горизонтальной, диагональных осей. Пример кроссворда с симметричной сеткой представлен на рисунке 4.



Рисунок 4 – Пример кроссворда с симметричной сеткой

* 1. Описание систем-аналогов

Программа, предназначенная для составления и разгадывания кроссворда, может быть выполнена в виде программы, которую можно установить на компьютер, либо в виде веб-приложения. Также подобный продукт может быть представлен мобильным приложением для смартфона.

Рассмотрим возможности известных программ, предоставляющих инструменты для создания и решения кроссвордов.

* + 1. Приложение «Сrosswordus»

Для создания кроссвордов можно использовать веб-приложение «Crosswordus» [8]. Сервис предоставляет большой выбор классических и фигурных сеток различных размеров, который постоянно пополняется.

Для создания можно воспользоваться генераторами кроссвордов. Генераторы позволяют автоматически создавать сетку с заданными параметрами, располагать введённые с клавиатуры слова на сетке, заполнять сетку словами из словаря.

В системе существует несколько способов создания кроссвордов:

* Создать кроссворд из заданных слов;
* Нарисовать сетку кроссворда и заполнить ее словами из словаря.

Созданным кроссвордом можно поделиться с другими пользователями сервиса или друзьями в социальных сетях, либо распечатать и разгадывать в бумажном варианте.

На рисунке 5 приведена главная экранная форма программы «Crosswordus», на которой можно заметить основные функции для создания кроссворда.

Рисунок 5 – Экранная форма сайта «Crosswordus»

* + 1. Приложение «Puzzlecup»

Puzzlecup – еще один сервис, который помогает без труда составлять кроссворды [8].

На рисунке 6 приведена главная экранная форма программы «Puzzlecup». На форме присутствует рабочая область, в которой пользователь с помощью мышки выделяет место для очередного слова и выбирает автоматически подобранные слова из словаря. Также можно задавать свои слова, заранее придумав определения к ним.

Составленный кроссворд можно сохранить и распечатать. Сохраняется он в куки браузера.

Рисунок 6 – Экранная форма программы «Puzzlecup»

* + 1. Приложение «Hot Potatoes»

«Hot Potatoes» – инструментальная программа-оболочка, предоставляющая преподавателям возможность самостоятельно создавать интерактивные задания, кроссворды и тесты для контроля и самоконтроля учащихся без знания языков программирования и привлечения специалистов в области программирования [9].

С помощью программы можно создать 10 типов упражнений и тестов по различным дисциплинам с использованием текстовой, графической, аудио- и видеоинформации.

Особенностью этой программы является то, что созданные задания сохраняются в стандартном формате веб-страницы: для их использования ученикам необходим только веб-браузер (например, Internet Explorer).

Ученикам не нужна программа «Hot Potatoes», она требуется только преподавателям для создания и редактирования упражнений.

Программа широко используется во всем мире для создания заданий для изучения любых дисциплин.

В состав «Hot Potatoes» входят 5 блоков программ для составления заданий и тестов разных видов. Один из блоков может быть использован как программа для создания кроссвордов. Он называется «JCross» [10].

На рисунке 7 приведена главная экранная форма программы «JCross», в которой существуют два режима:

* ручное размещение слов;
* автоматическое размещение слов.

  
Рисунок 7 – Экранная форма программы «Hot Potatoes: JCross»

При ручном размещении слов положение слов в сетке определяется пользователем: слова буква за буквой вводятся в поле кроссворда. При автоматическом размещении слов все слова вводятся списком в поле ввода слов в открывшемся диалоге, при этом все слова вводятся в отдельные строки, а затем программа подбирает варианты их размещения. В поле Максимальный размер указывается максимальное число ячеек в кроссворде. С помощью кнопок-стрелок в левом нижнем углу экрана можно перемещать созданный кроссворд в рамках поля. Распечатать кроссворд можно в двух вариантах: незаполненный и заполненный.

* 1. Диаграмма объектов предметной области

Объектно-ориентированный анализ и проектирование – технология разработки программных систем, в основу которых положена объектно-ориентированная методология представления предметной области в виде объектов, являющихся экземплярами соответствующих классов [11].

Главными объектами в системе являются кроссворды, которые будет разгадывать пользователь. Кроссворды создаются и редактируются администратором и хранятся в файлах. Кроссворды состоят из слов, которые образуют сетку, и заданий к ним, которые состоят из определений. Слова и определения хранятся в словаре понятий. Пользователь разгадывает кроссворд, после чего может проверить правильность своих ответов. На рисунке 7 приведена диаграмма объектов предметной области.

Рисунок 7 – Диаграмма объектов предметной области

* 1. Постановка задачи

Перед авторами поставлена задача – разработать систему составления и разгадывания классического кроссворда с функциями администратора. Система должна представлять собой настольное приложение. В ней должны быть предусмотрены две роли пользователей: администратор и пользователь.

Для работы в системе пользователь должен авторизоваться в ней, вводя логин и пароль. После этого система должна проверить, зарегистрирован ли пользователь и корректен ли введённый пароль. Если пользователь с указанным логином не зарегистрирован, или введен неверный пароль, то система должна выдать соответствующее сообщение. В зависимости от роли пользователю будут доступны различные функции.

*Режим администратора*

Администратору должны быть доступны следующие функции.

1. *Создание кроссворда*. Перед созданием кроссворда администратор должен указать его параметры (высота от 7 до 30 клеток, ширина в пределах от 7 до 30 клеток и выбрать словарь). Создание нового кроссворда может производиться в одном из двух режимов: автоматическом или ручном.

* *Автоматический режим.*  В этом режиме генерация кроссворда производится по правилу максимального заполнения сетки.
* *Ручной режим.*  В этом режиме администратор заполняет кроссворд с помощью инструментов для создания кроссвордов: добавления, удаления и изменения слова, изменения ориентации слова. Администратору должно быть предоставлено использование словаря. Размещение слов будет возможно либо определением места слова на сетке, либо с использованием технологии «Drag-and-drop».

При необходимости администратор может сохранить кроссворд в файл.

1. *Работа со словарями понятий*. Система должна предоставить администратору возможность создания нового словаря понятий, загрузки словаря и редактирования его. Администратор должен иметь возможность добавления слова, изменять их и удалять. При этом система должная проверить уникальность понятий и проверить их правильность написания на русском языке. В системе должна быть реализована два вида сортировок словаря по алфавиту (от «А» до «Я», от «Я» до «А») и по длине (по возрастанию, по убыванию). А так же система должна предоставить доступ к поиску по маске.

*Режим пользователя*

Пользователю должно быть доступно толькоразгадывание кроссворда. Система должна предоставить пользователю возможность открыть из файла кроссворд для решения. Разгадывание кроссворда должно вестись непосредственно на сетке: пользователь должен побуквенно вводить слова, удалять отдельные буквы, при разгадывании должна быть организована система подсказок. Система должна проверять правильность разгадывания. Так же для сохранения промежуточного результата решения кроссворда, должна быть возможность сохранения кроссворда в файл.

Таким образом, система должна выполнять следующие функции:

1. *общесистемные функции:*
   * авторизация 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;
   * контроль количества взятых подсказок;
   * проверка правильности разгадывания кроссворда;
   * выдача справочной информации о системе.
2. *функции администратора:*
   * настройка параметров кроссворда при создании:
3. задание высоты;
4. задание ширины;
5. подключение словаря понятий;
   * составление/редактирование кроссворда:
6. добавление слова;
7. удаление слова;
8. изменение ориентации слова;
9. фильтрация словаря понятий по маске;
10. сортировка словаря
    * сохранение кроссворда в файл заданной структуры;
    * загрузка кроссворда из файла;
    * работа со словарями понятий:
11. добавление понятия;
12. удаление понятия;
13. изменение понятия;
14. проверка дублирования понятий;
15. проверка языка записи понятий;
16. сортировка словаря по выбранному критерию;
17. поиск по заданной маске;
18. загрузка словаря из файла;
19. сохранение словаря из файла;

10) создание нового словаря понятий;

1. *функции пользователя:*
   * загрузка кроссворда из файла;
   * разгадывание кроссворда с организацией системы подсказок:
2. выбор слова;
3. выписывание/удаление/изменение буквы;
4. взятие подсказки
   * сохранение кроссворда в файл.
5. Проектирование системы
   1. Структурная схема системы

Под системой понимается единство целей, ресурсов и строения (структуры), свойства которого определяются отношение входящих в рассматриваемую совокупность объектов (элементов).

*Структурная схема* – это совокупность элементарных звеньев объекта и связей между ними, один из видов графической модели, которая разрабатывается на начальных стадиях проектирования. Она определяет основные функциональные части, назначение и взаимосвязи между ними [12].

При разработке структурной схемы используется методология структурного проектирования, в основе которой лежит алгоритмическая декомпозиция и иерархия вида «часть-целое», учитывающая, что внутренние связи элементов внутри подсистем сильнее, чем связь между подсистемами. Декомпозиция системы может повторяться многократно, вплоть до уровня конкретных процедур, при этом должна быть обеспечена целостность системы, а все составляющие компоненты взаимоувязаны. Для этого используются такие принципы разработки, как «сверху-вниз», «иерархическое упорядочивание» и другие [13].

Структурная схема состоит из одной большой части и изображена на рисунке 8. В состав систем входят следующие подсистемы:

1. *подсистема авторизации*, которая определяет роль пользователя и настраивает интерфейс;
2. *подсистема «Администратор»*, в состав которой входят:

* *подсистема ручного создания кроссворда*, позволяющая добавлять, удалять слова, изменять их ориентацию;
* *подсистема генерирования кроссворда*, позволяющая сгенерировать кроссворд на основе заданных параметров;
* *подсистема настройки параметров*, которая отвечает за выбор значений параметров кроссворда и поверку корректности этих значений, а также *подсистема работы со словарем,* которая позволяет добавлять, удалять, изменять слова из словаря понятий.

1. *подсистема «Пользователь»*, которая позволяет разгадать кроссворд;
2. *файловая подсистема*, позволяющая загружать и сохранять кроссворд в файл;
3. *подсистема визуализации*, которая отображает процесс работы с кроссвордом на этапе его создания и разгадывания;
4. *справочная подсистема*, которая выдает сведения о системе (руководство пользователя) и об ее разработчиках.



Рисунок 8 – Структурная схема системы

* 1. Спецификация системы

*Спецификация* *системы* – это документ, который содержит полное и четкое описание разрабатываемого продукта [14]. В терминологии программной инженерии ее называют «разработкой требований» (спецификацией требований).

*Требования* – это свойства, которыми должно обладать программное обеспечение для адекватного задания необходимых функций, а также условия и ограничения на программное обеспечение, данные, среду выполнения и технику [14]. Требования отражают потребности людей (заказчиков, пользователей, разработчиков), заинтересованных в создании программного обеспечения. Различают требования программные, системные, а также функциональные и нефункциональные требования.

*Программные требования* определяют требования к процессу, операционной системе, режиму выполнения ПО, выбору платформы, а также к ним относятся извлечение (сбор), специфицирование и утверждение требований к разрабатываемой программной системе [14].

*Системные требования* описывают требования к программной системе, состоящей из взаимосвязанных программных и аппаратных подсистем и разных приложений [14].

*Функциональные требования* задают назначения системы, а *нефункциональные* – условия выполнения программного обеспечения (например, скорость отклика при выполнении заданной операции) [14].

К нефункциональным требованиям относятся защита данных, адаптивность, изменчивость, которые в свою очередь описываются атрибутами качества (*спецификация качества*) [14].

* + 1. Функциональная спецификация

Функциональная спецификация – это документ, описывающий требуемые характеристики системы (функциональность) [15]. Документация описывает необходимые для пользователя системы входные и выходные параметры. Спецификации помогают устранить дублирование и несоответствия, позволяют точно оценить необходимые действия и ресурсы, выступают в качестве согласующего и справочного документов о внесённых изменениях, предоставляют документацию с конфигурацией, и даёт возможность взаимодействия лиц, работающих с основными функциями системного проектирования. Они дают точное представление о решении проблемы, повышая эффективность разработки системы и оценивая стоимость альтернативных путей проектирования [16].

Функциональная спецификация не определяет операции, происходящие внутри данной системы и каким образом будет реализована её функция. Вместо этого, она рассматривает взаимодействие с внешними агентами (например, персонал, использующий программное обеспечение; периферийные устройства компьютера или другие компьютеры). Функциональная спецификация должна в полном объёме отображать информационные связи проектируемой системы как с внешним миром, так и между подсистемами.

Функциональная спецификация системы приведена в таблице 2.1.

* + 1. Спецификация качества

*Качество программного обеспечения –* способность [программного продукта](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) при заданных условиях удовлетворять установленным или предполагаемым [потребностям](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D1%80%D0%B5%D0%B1%D0%BD%D0%BE%D1%81%D1%82%D1%8C) [17].Онаописывают дополнительные характеристики продукта в различных «измерениях», важных для пользователей и/или разработчиков, касается вопросов портируемости, прозрачности взаимодействия с другими системами, целостности, устойчивости и т.п.

Таблица 1 – Перечень функций, выполняемых системой

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название  подсистемы | Название функции | Информационная среда | | | |
| Входные данные | | Выходные данные | |
| Назначение (наименование) | Тип, ограничения | Назначение (наименование) | Тип, ограничения |
|  |  |  |  |  |  |
| 1 Справочная | 1.1 Выдать сведения о разработчиках | Сведения о разработчиках системы (ФИО, номер группы) | Текст (МЕМО) | Визуальное отображение информации | ‑ |
| 1.2 Выдать сведения о системе | Файл справки | Текстовый (\*.HTML) |
| Код ошибки | целое |
| 2 Настройки параметров | 2.1 Подключить словарь понятий | Имя файла | Строка, \*.dict | Список понятий и их определений | Объект «Словарь» |
| Код ошибки | Целое |
| 2.2 Задать ширину кроссворда | Диапазон допустимых значений | Целое, 7..20 | Ширина кроссворда | Целое |
| 2.3 Задать высоту кроссворда | Диапазон допустимых значений | Целое, 7..20 | Высота кроссворда | Целое |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| 3 Генерирования кроссворда | Сгенерировать кроссворд по заданным параметрам | Ширина | Целое | Кроссворд | Объект «Кроссворд» |
| Высота | Целое |
| Список понятий и определений | Объект «Словарь» |
| 4 Создания кроссворда | 4.1 Выделить область на сетке | Кроссворд | Объект «Кроссворд» | Маска | Строка [3..20] |
| 4.2 Выбрать слово для добавления | Список понятий | Динамический массив строк | Выбранное понятие из словаря | Строка [3..20] |
| 4.3 Добавить слово | Выбранное понятие из словаря | Строка [3..20] | Кроссворд | Объект «Кроссворд» |
| 4.4 Выбрать слово для удаления | Список добавленных понятий | Динамический массив строк | Выбранное понятие из списка добавленных | Строка [3..20] |
| 4.5 Удалить слово | Выбранное понятие из списка добавленных | Строка [3..20] | Кроссворд | Объект «Кроссворд» |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 4 Создания кроссворда | 4.6 Отфильтровать словарь понятий по маске | Список понятий | Динамический массив строк | Отфильтрованный список понятий | Динамический массив строк |
| Маска | Строка [1..20] |
| 4.7 Отсортировать словарь понятий | Список понятий | Динамический массив строк | Отсортированный список понятий | Динамический массив строк |
| Вид сортировки | Перечислимый |
| 5 Работы со словарем | 5.1 Ввести понятие | Набор символов | Кириллица | Понятие | Строка [3..20] |
| 5.2 Ввести определение | Набор символов | Кириллица, цифры, специальные символы | Определение | Строка [100] |
| 5.3 Добавить понятие | Список понятий и определений | Объект «Словарь» | Измененный список понятий и определений | Объект «Словарь» |
| Понятие | Строка [3..20] |
| Определение | Строка [100] |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 5 Работы со словарем | 5.4 Выбрать понятие для изменения | Список понятий и определений | Объект «Словарь» | Выбранное понятие и определение | Объект «Словарь» |
| 5.5 Изменить понятие и/или определение | Выбранное понятие и определение | Объект «Словарь» | Измененный список понятий и определений | Объект «Словарь» |
| Понятие | Строка [3..20] |
| Определение | Строка [100] |
| 5.6 Выбрать понятие для удаления | Список понятий и определений | Объект «Словарь» | Выбранное понятие и определение | Объект «Словарь» |
| 5.7 Удалить понятие | Выбранное понятие и определение | Объект «Словарь» | Измененный список понятий и определений | Объект «Словарь» |
| 5.8 Проверить дублирование понятий | Список понятий и определений | Объект «Словарь» | Признак дублирования | Логическое |
| Понятие | Строка [3..20] |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 5 Работы со словарем | 5.9 Проверить язык записи понятий | Список понятий и определений | Объект «Словарь» | Признак языка | Перечислимый |
| 5.10 Выбрать критерий сортировки | Список критериев | По алфавиту | Выбранный критерий сортировки | Перечислимый |
| По длине |
| 5.11 Отсортировать словарь по критерию | Список понятий и определений | Объект «Словарь» | Отсортированный список понятий и определений | Объект «Словарь» |
| Выбранный критерий сортировки | Перечислимый |
| 5.12 Отфильтровать словарь понятий по маске | Список понятий и определений | Объект «Словарь» | Отфильтрованный список понятий и определений | Объект «Словарь» |
| Маска | Строка [1..20] |
| 5.13 Загрузить словарь из файла | Имя файла | Строка, \*.dict | Список понятий и определений | Объект «Словарь» |
| 5.14 Сохранить словарь в файл | Список понятий и определений | Массив строк | Файл | \*.dict |
| Имя файла | Строка, \*.dict |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 6 Файловая | 6.1 Сохранить кроссворд в файл | Кроссворд, имя файла | Объект «Кроссворд», строка, \*.cswd | Файл | Структура файла определяется в ходе проектирования |
| 6 Файловая | 6.2 Загрузить кроссворд из файла | Имя файла | Строка, \*.cswd | Кроссворд | Объект «Кроссворд» |
| 7 Визуализации | 7.1 Отобразить кроссворд | Кроссворд | Объект «Кроссворд» | Отображение сетки, отображение заданий | – |
| 7.2 Выделить слово на сетке | Выбранное определение из списка заданий | Строка [3..20] | Визуализация отображения слова | – |
| 7.3 Отобразить количество подсказок | Количество заданий | Целое | Отображение количества подсказок | – |
| 7.4 Отобразить подсказку | Выбранное определение из списка заданий | Динамический массив строк | Отображение слова на сетке | – |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 8 Разгадывания кроссворда | 8.1 Выбрать определение | Список определений | Динамический массив строк | Выбранное определение | Строка [100] |
| 8.2 Вписать букву | Буква | Символ | Кроссворд | Объект «Кроссворд» |
| Координаты буквы | Целое |
| 8.3 Взять подсказку | Выбранное определение из списка заданий | Динамический массив строк | Кроссворд | Объект «Кроссворд» |
| Количество подсказок | Целое, >0 | Количество подсказок | Целое |
| 8.4 Проверить правильность разгадывания | Кроссворд | Объект «Кроссворд» | Результат разгадывания | Логическое |
| 9 Регистрации | 9.1 Ввести логин | Набор символов | Строка [5..15] | Логин | Строка [5..15] |
| Латиница |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 9 Регистрации | 9.2 Ввести пароль | Набор символов | Строка | Пароль1 | Строка [5..15] |
| Латиница + цифры |
| 9.3 Повторить ввод пароля | Набор символов | Строка | Пароль2 | Строка [5..15] |
| Латиница + цифры |
| 9.4 Проверить совпадение паролей | Пароль1 | Строка [5..15] | Признак совпадения паролей | Логическое |
| Пароль2 | Строка [5..15] |
| 9.5 Проверить уникальность логина | Логин | Строка [5..15] | Признак совпадения логинов | Логическое |
| Список зарегистрированных пользователей | Сущность БД «Пользователь» |
| 10 Авторизации | 10.1 Ввести логин | Набор символов | Строка [5..15], латиница | Логин | Строка |
| 10.2 Ввести пароль | Набор символов | Строка [5..15], латиница + цифры | Пароль | Строка |
| 10.3 Проверить учетную запись | Логин, пароль, список учетных записей | Строка [5..15], строка [5..15], сущность БД | Роль пользователя, код ошибки | Перечислимое, целое |

В системе должны выполняться следующие требования:

− защита от необдуманных действий пользователя;

− контроль вводимых параметров должен осуществляться с учетом заданного диапазона значений;

− проверка структуры файла, хранящего сведения о кроссворде;

– требования к разработке интерфейса;

− корректное выполнение функций, поставленных перед системой.

Спецификация качества включает в себя перечень исключительных ситуаций и описание реакций системы на них [18].

* + 1. 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 [19].

В таблице 2 приведен перечень исключительных ситуаций для разрабатываемой системы и описаны реакции системы на их возникновение.

Таблица 2 – Перечень исключительных ситуаций

|  |  |  |
| --- | --- | --- |
| Название  подсистемы | Название исключительной ситуации | Реакция системы |
| 1 | 2 | 3 |
| 1 Справочная | 1.1 Невозможно открыть файл справки | Выдача сообщения «Файл справки поврежден» |
| 1.2 Невозможно найти файл справки | Выдача сообщения «Отсутствует файл справки» |

Продолжение таблицы 2

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 2 Файловая | 2.1 Попытка открытия файла с несобственным форматом | Выдача сообщения «Файл поврежден или недопустимого формата» |
| 2.2 Файл с заданным именем не существует | Выдача аналогичного сообщения |
| 3 Авторизации | 1.1 Комбинация имени пользователя и пароля отсутствует | Выдача сообщения «Неверное имя пользователя или пароль» |
| 4 Работы со словарем понятий | 4.1 Нет файла с таким именем | Выдача сообщения «Файл словаря понятий не найден» |
| 4.2 Структура файла изменена | Выдача сообщения «Неверная структура файла » |
| 6 Настройки параметров | 6.1 Файл словаря понятий отсутствует | Выдача сообщения «Не удалось загрузить словарь» |
| 6.2 Файл словаря понятий не соответствует структуре | Выдача аналогичного сообщения |

* + 1. Разработка прототипа интерфейса пользователя системы

*Пользовательский интерфейс* – совокупность элементов, позволяющих пользователю управлять работой программы или вычислительной системы и получать требуемые результаты.

Разработка пользовательского интерфейса включает следующие основные этапы [20]:

* 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 [20].

Прототип интерфейса пользователя представлен на рисунках 9-21.

На рисунке 9 представлен прототип экранной формы авторизации, где пользователь должен ввести логин и пароль. После этого система должна проверить введенные данные и настроить интерфейс в соответствии с ролью: пользователь или администратор.

   
Рисунок 9 – Прототип экранной формы окна авторизации

Прототип экранной формы регистрации представлен на рисунке 10, где пользователь, если он не зарегистрирован, должен ввести свой логин и пароль, затем подтвердить пароль повторным вводом.

   
Рисунок 10 – Прототип экранной формы окна регистрации

  
Рисунок 11 – Прототип экранной формы окна администратора

На рисунке 11 представлен прототип главного меню администратора. При нажатии на кнопку «Создать кроссворд» должно открыться окно «Создание кроссворда» (рисунок 12).

   
Рисунок 12 – Прототип экранной формы окна создания кроссворда

Здесь администратор должен выбрать ширину кроссворда в поле «Ширина сетки», длину кроссворда в поле «Высота сетки» и выбрать словарь в поле «Словарь понятий» с помощью кнопки «Обзор». После этого необходимо нажать кнопку «Сгенерировать» или «Создать вручную». При нажатии на кнопку «Сгенерировать» программа должна автоматически создать сетку кроссворда, подбирая слова из словаря понятий. При выборе ручного создания кроссворда должно открываться окно «Создание/редактирование кроссворда» (рисунок 13).

Рисунок 13 – Прототип экранной формы окна  
создания/редактирования кроссворда

Перед добавлением слова нужно выделить область на сетке, которую нужно заполнить, в результате на список понятий будет наложен фильтр, соответствующий маске, нарисованной на сетке. Администратор должен выбрать слово из списка и нажать на кнопку C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\+.png. Удаление должно происходить следующим образом: администратору нужно выбрать понятие из списка добавленных и нажать на кнопку C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\-.png. Также на форме должны присутствовать инструменты работы со словарем, осуществляющие поиск, сортировку по алфавиту и длине. Кнопка C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\save.png должна открывать окно «Сохранения кроссворда» (рисунок 14), в котором необходимо указать путь к файлу, который будет хранить кроссворд. Кнопка C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\back.png должна возвращать администратора в окно «Режим администратора».

   
Рисунок 14 – Прототип экранной формы окна сохранения кроссворда

При нажатии кнопки «Редактировать кроссворд» в режиме администратора должно открываться окно «Открыть кроссворд» (рисунок 15), в котором с помощью кнопки «Обзор» необходимо выбрать нужный файл. При нажатии на «ОК» должен осуществиться переход на форму «Создание/редактирование кроссворда» (см. рисунок 13).

  
Рисунок 15 – Прототип экранной формы окна открытия кроссворда

Если администратор нажал на кнопку «Создать словарь понятий», система должна открыть форму «Создание/редактирование словаря понятий» (рисунок 16).

При нажатии на кнопку C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\++.png или C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\editDict.png должно открываться окно «Добавление/изменение понятия» (рисунок 17), в котором осуществляется добавление нового или изменение уже имеющегося понятия соответственно.

После сохранения изменений они должны отобразиться в таблице на форме «Создание/редактирование словаря понятий». Для удаления понятия и его определения администратору нужно выбрать соответствующую строку в

  
Рисунок 16 – Прототип экранной формы окна   
создания/редактирования словаря

   
Рисунок 17 – Прототип экранной формы окна добавления/изменения понятия

таблице и нажать на кнопку C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\delete.png. Кнопки C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\sortN.png и C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\sortA.png должны сортировать список по длине и алфавиту соответственно. Для фильтрации списка необходимо вписать маску в строку поиска. Для сохранения изменений нужно нажать кнопку C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\save.png, после чего откроется окно «Сохранить словарь понятий» (рисунок 18).

   
Рисунок 18 – Прототип экранной формы окна сохранения словаря понятий

При нажатии кнопки «Редактировать словарь» в окне «Режим администратора» должно открываться окно «Открыть словарь» в котором с помощью кнопки «Обзор» выберется нужный файл (рисунок 18). При нажатии на «ОК» откроется форма «Создание/редактирование словаря» (рисунок 16).



Рисунок 18 – Прототип экранной формы окна открытия словаря понятий

В режиме пользователя сразу после регистрации или входа в систему должно открываться окно «Открыть кроссворд» (рисунок 15). Пользователю нужно указать путь к файлу, содержащему желаемый кроссворд, и нажать «ОК». После этого осуществляется переход на форму «Разгадывание кроссворда» (рисунок 19).

После перехода к окну «Разгадывание кроссворда» пользователь приступает к решению кроссворда. Сначала необходимо выбрать определение из списка заданий, на сетке должна подсветиться область, соответствующая описываемому определением понятию, в эту область нужно вписывать буквы. Пользователю должна быть доступна возможность воспользоваться подсказкой. Для этого ему необходимо выбрать определение, вызвавшее трудность, и нажать на кнопку C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\help.png, после чего в

Рисунок 19 – Прототип экранной формы окна разгадывания кроссворда

подсвеченной области на сетке отобразится правильное слово. После окончания разгадывания кроссворда необходимо нажать кнопку C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\check.png. Если кроссворд решен верно, должна появится иконка C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\up.png, иначе C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\down.png. Также необходимо предоставить возможность пользователю сохранить нерешенный кроссворд при нажатии на кнопку C:\Users\User\Desktop\ТП\Прототипы UI\Рисунки\save.png.

При нажатии на кнопку «Справка» в окне авторизации должно открываться окно «Информация» (рисунок 20), на котором должна быть представлена информация о системе и ее разработчиках.

  
Рисунок 20 – Прототип экранной формы окна информации

На рисунке 21 приведена навигационная модель приложения.

Рисунок 21 – Навигационная модель приложения

* 1. Разработка структур данных и классов

Для представления абстракций объектов используется специальный определяемый программистом тип данных – класс.

Класс – это структурный тип данных, который включает описание полей данных, а также процедур и функций, работающих с этими полями данных [18].

Диаграммы классов – это наиболее часто используемый тип диаграмм, которые создаются при моделировании объектно-ориентированных систем, они показывают набор классов, интерфейсов и коопераций, а также их связи [18].

На рисунке 22 изображена диаграмма классов, которая показывает отношения классов между собой.



Рисунок 22 – Диаграмма классов системы

В таблицах 3-7 представлены описания всех классов. Они необходимы для понимания назначения сущностей, входящих в диаграмму классов системы.

Таблица 3 – Описание класса «Пользователь»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип | Описание |
| логин | Строка | Логин пользователя |
| пароль | Строка | Пароль пользователя |
| роль | Булево значение | Роль пользователя |

Таблица 4 – Описание класса «Кроссворд»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип | Описание |
| сетка | Тип Сетка | Сетка |
| словарь | Тип Словарь | Словарь понятий |
| вертикальноезадание | Строка | Задание по вертикали |
| горизонтальноезадание | Строка | Задание по горизонтали |

Таблица 5 – Описание класса «Сетка»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип | Описание |
| ширина | Целое | Ширина кроссворда |
| высота | Целое | Высота кроссворда |
| матрицаячеек | Матрица типа Сетка | Матрица ячеек |

Таблица 6 – Описание класса «Ячейка»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип | Описание |
| буква | char | Буква в ячейке |

Таблица 7 – Описание класса «Словарь»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип | Описание |
| ключ | Строка | ключ |
| значение | Строка | значение |

* 1. Разработка и описание алгоритмов обработки данных

Понятие алгоритма обработки данных используется для описания метода решения задачи, который в дальнейшем возможно реализовать в выбранной среде программирования [19].

Алгоритм – набор инструкций, описывающих порядок действий исполнителя для достижения некоторого результата [20].

Диаграмма деятельности в режиме «Администратор» показана на рисунке 24. Первоначально необходимо будет выбрать один из режимов – «Создать кроссворд», «Редактировать кроссворд», «Создать словарь понятий» или «Редактировать словарь понятий». При выборе первого режима администратор может задать параметры будущего кроссворда. При выборе второго открывается окно для открытия кроссворда из файла, а после окно с функциями для редактирования. При выборе третьего открывается окно с набором функций для создания словаря понятий. При выборе четвертого открывается окошко для загрузки словаря из файла, и в последствии окно для его редактирования. После всего нужно сохранить словарь понятий или кроссворд.

Диаграмма деятельности в режиме «Пользователь» показана на рисунке 23 Для начала пользователю необходимо будет выбрать из файла кроссворд для разгадывания. Потом следует процесс разгадывания с функциями: выбор определения, чтобы заполнить его понятие на сетке, редактировать букву на сетке, взять подсказку и проверить правильность решения кроссворда. В процессе разгадывания доступна функция сохранить кроссворд в файл и вернуться назад.



Рисунок 23 – Диаграмма деятельности пользователя

  
Рисунок 24 – Диаграмма деятельности администратора

* 1. Выбор и обоснование комплекса программных средств
     1. Выбор языка программирования и среды разработки

Выбранным языком программирования является C#. Это язык программирования для разработки приложений, предназначенных для выполнения в среде .NET Framework. Язык C# прост: он многофункционален, гибок и хорошо адаптирован под операционные системы, на которых работают 95% компьютеров во всем мире. По сравнению с другими языками программирования он более удобен и прост в использовании [21].

Язык C# является основным языком программирования платформы .NET, и владение им открывает доступ программисту к широкому спектру технологий, поддерживаемых платформой .NET [22]. С помощью С# можно разрабатывать:

* любые приложения для настольных компьютеров (Windows Forms);
* веб-приложения любой сложности для интернета (ASP.Net MVC, MVVM);
* серверные технологии доступа к данными (ADO.NET, LINQ, Entity Framework);
* распределённые системы на основе веб-сервисов;
* приложения для планшетов (Windows 10);
* приложения для смартфонов (Windows Phone);
* кроссплатформенные приложения для операционных систем iOS, Android (VS 2015, Xamarin);
* игры для Windows, X-Box (Unity3D);
* приложения облачных технологий (Windows Azure);
* приложения для встраиваемых систем (интернет вещей).
  + 1. Выбор операционной системы

В качестве операционной системы была выбрана система MS Windows 10. Windows 10 – операционная система для персональных компьютеров и рабочих станций, разработанная корпорацией Microsoft в рамках семейства Windows NT. Она поддерживает MS Visual Studio 2017 и предоставляет разработчикам все необходимые средства для создания качественного приложения.

* + 1. Выбор среды программирования

В качестве среды разработки была выбрана MS Visual Studio 2017. Microsoft Visual Studio предоставляет несколько способов разработки приложений Windows, работающих локально на компьютерах пользователей. Преимущество использования Visual Studio заключается в том, что эта среда предоставляет средства, благодаря которым процесс разработки приложений становится намного быстрее, проще и надежнее [23].

В Visual Studio 2017 есть современные средства и службы, которые можно использовать для создания уникальных приложений для устройств, облачных инфраструктур и множества других сред [24].

1. Реализация системы
   1. Разработка и описание интерфейса пользователя

Интерфейс пользователя – совокупность средств и методов, при помощи которых пользователь взаимодействует с различными, чаще всего сложными, машинами, устройствами и аппаратурой [25].

После запуска системы откроется окно авторизации, предназначенное для входа в систему. В данном окне пользователь может выполнить вход в систему или зарегистрироваться как новый пользователь, нажав на кнопку «Регистрация».

В системе предусмотрена работа двух типов пользователей: администратор (рисунок 25) и пользователь.

  
Рисунок 25 – Форма авторизации

Для регистрации нового пользователя необходимо ввести логин, пароль и подтверждение пароля (рисунок 26).

Нажав на кнопку «Информация» пользователь увидит форму   
(рисунок 27), в которой увидит сведения о системе и её авторах.

* + 1. Режим пользователя

После авторизации пользователь должен загрузить кроссворд (рисунок 28), чтобы начать или продолжить разгадывание кроссворда. Для этого нужно выбрать файл с расширением .crwd и открыть его. После появляется окно для разгадывания кроссворда (рисунок 29).

  
Рисунок 26 – Форма для регистрации нового пользователя

  
Рисунок 27 – Форма информации системы

* + 1. Режим администратора

После авторизации пользователь переходит в меню администратора (рисунок 30), ему доступны следующе функции: «Создать кроссворд», «Редактировать кроссворд», «Создать словарь понятий», «Редактировать словарь понятий».

При нажатии на кнопку «Создать кроссворд» осуществляется переход к форме «Создание кроссворда» (рисунок 31), где можно задать параметры кроссворда и выбрать словарь понятий. После нажатия кнопки «Создать кроссворд» появляется форма «Создание/редактирование кроссворда» (рисунок 32), в которой есть инструменты для создания кроссворда.

  
Рисунок 28 – Загрузка кроссворда

  
Рисунок 29 – Экранная форма для разгадывания

  
Рисунок 30 – Главное меню администратора

  
Рисунок 31 – Экранная форма для задания параметров кроссворда

При выборе пункта меню «Редактировать кроссворд» откроется форма, в которой надо загрузить кроссворд из файла с расширением .crwd. Откроется форма, которая представлена на рисунке 32, на котором можно составить кроссворд. На этой форме администратор может сгенерировать кроссворд или создать кроссворд вручную. Для генерации в автоматическом режиме необходимо нажать на . При добавлении слова в ручном режиме администратор должен выделить область, система по маске должна отфильтровать слова в словаре, после этого администратор сможет добавить слово на сетку, щёлкнув двойным кликом мыши на нужное слово. Администратор может удалить слово из сетки, выделив слово и нажав кнопку . Так же администратор может открыть новый словарь понятий и работать с ним.

  
Рисунок 32 – Экранная форма для создания кроссворда

При выборе пункта меню «Создать словарь понятий» или «Редактировать словарь понятий» откроется форма (рисунок 33), где администратор может работать со словарем: добавлять слова, удалять слова и редактировать, выполнять сортировку (по длине и по алфавиту). Система должна контролировать язык записи понятий и их уникальность.

  
Рисунок 33 – Экранная форма создания или редактирования кроссворда

* + 1. Описание тестового примера

Создадим новый кроссворд в режиме администратора с параметрами: длина сетки – 10, ширина сетки– 10, – и заполним его понятиями из словаря, как показано на рисунке 34. Затем сохраним его в файл с название «test.crwd», как показано на рисунке 35.

  
Рисунок 34 – Создание нового кроссворда в режиме администратора

  
Рисунок 35 – Сохранение кроссворда в файл

Перейдем в режим пользователя. Ему необходимо открыть файл, в котором составлен кроссворд. На рисунке 36 видно, что открылся кроссворд, созданный в режиме администратора.

  
Рисунок 36 – Экранная форма открыт для разгадывания кроссворд

Процесс разгадывания кроссворда на форме представлен на рисунке 37. Чтобы разгадать кроссворд необходимо сначала выбрать вопрос, щелкнув по заданию левой кнопкой мыши, система выделит на сетке слово, которое необходимо вписать побуквенно. Так же можно сохранить кроссворд во время разгадывания. Во время разгадывания можно использовать подсказку. Для этого пользователь должен выделить задание и нажать на кнопку лампочка (см. рисунок 38). Система уменьшит количество подсказок на единицу. Данная кнопка будет недоступна, когда количество подсказок будет равно нулю.

По завершению разгадывания кроссворда пользователь видит соответствующее сообщение, как показано на рисунке 39.

  
Рисунок 37 – Процесс разгадывания кроссвода

  
Рисунок 38 – Использование подсказки

  
Рисунок 39 – Завершение разгадывания кроссворда

* 1. Реализация классов и структур данных

В соответствии со спецификацией, представленной в пункте 2.2, и выбранным языком программирования была получена диаграмма сущностных классов, представленная на рисунке 40.

Основные поля и методы классов описаны в таблицах 3.1-3.9.

Таблица 3.1 – Реализация класса «FormRegistration»

|  |  |
| --- | --- |
| Поля | SqlConnection sqlConnection – объект, который используется для соединения с базой данных |
| string login – логин пользователя |
| string pass – пароль пользователя |
| string passY – повторный пароль пользователя |
| Операции | Public FormRegistration (SqlConnection sqlConnection, FormMain formMain) – конструктор формы регистрации |
|  | Private void buttonOk\_Click () – записывает данные пользователя в базу данных |
|  | Private void buttonCancel\_Click () – закрывает форму регистрации |

Таблица 3.2 – Реализация класса «FormAdmin»

|  |  |
| --- | --- |
| Операции | Private void buttonClose\_Click () – закрывает форму режима администратора |
|  | Private void buttonOpenCreateCros\_Click () – открывает форму создания кроссворда |
|  | Private void buttonOpenCreateDict\_Click () – открывает форму создания словаря |
|  | Private void buttonOpenEditDict\_Click () – открывает форму редактирования словаря |
|  | Private void buttonOpenEditCros\_Click () – открывает форму редактирования кроссворда |
|  | Public FormAdmin (FormMain forMain) – конструктор формы режима администратора |

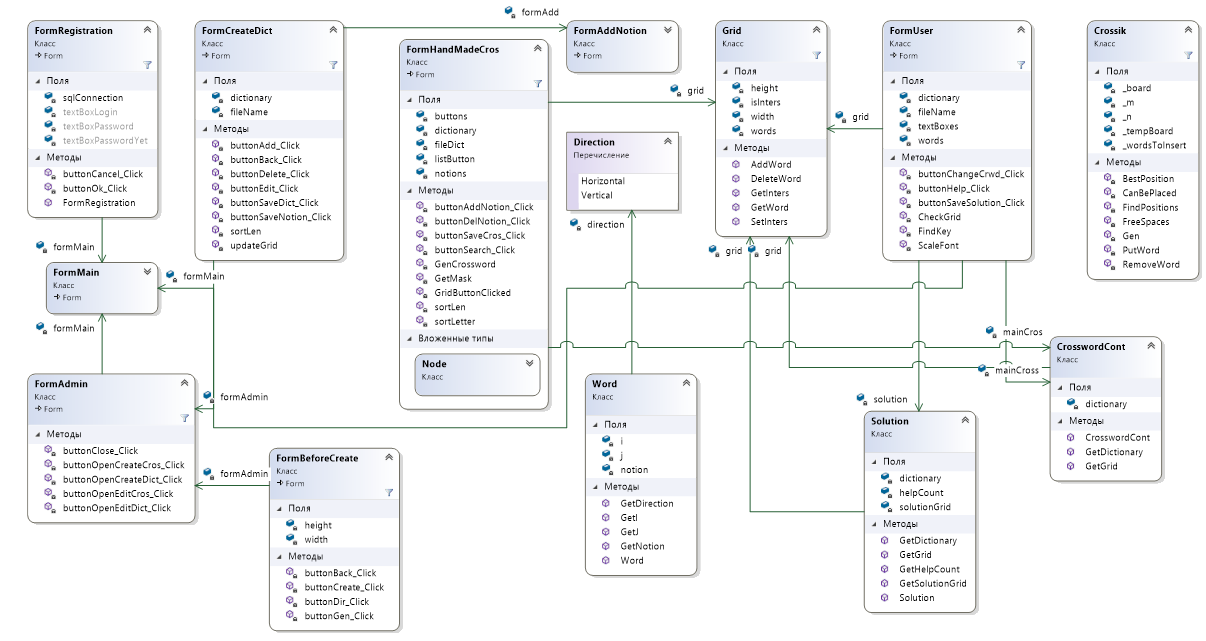


Рисунок 40 – Диаграмма классов систем

Таблица 3.3 – Реализация класса «FormBeforeCreate»

|  |  |
| --- | --- |
| Поля | Private int width – ширина кроссворда |
| Private int height– высота кроссворда |
| Операции | Private void buttonDir\_Click () – открывает окно с поиском словаря |
|  | Private void buttonBack\_Click () – возвращает назад на предыдущую форму |
|  | Private void buttonCreate\_Click () – создает кроссворд по заданным параметрам |

Таблица 3.4 – Реализация класса «FormCreateDict»

|  |  |
| --- | --- |
| Поля | Private Dictionary<string, string> dictionary – коллекция с определениями и понятиями словаря |
| Private string fileName – имя файла |
| Операции | Private void buttonAdd\_Click () – добавление понятия в словарь |
|  | Private void buttonDelete\_Click () – удаление выбранного определения и понятия из словаря |
|  | Private void buttonEdit\_Click () – редактирование определения и понятия в словаре |
|  | Private void buttonSaveDict\_Click () – сохранение измененного словаря |
|  | Private void sortLen () – сортировка слова из словаря по длине |
|  | Private void updateGrid () – изменение таблицы словаря |

Таблица 3.5 – Реализация класса «Word»

|  |  |
| --- | --- |
| Поля | Private int i – координата слова по x |
| Private int j – координата слова по y |
| Операции | Public Direction GetDirection () – получение направления слова |
|  | Public string GetNotion () – получение понятия слова |

Таблица 3.6 – Реализация класса «FormHandMadeCros»

|  |  |
| --- | --- |
| Поля | Private Button[ , ] buttons – матрица кнопок сетки |
| Private List<Button> listButton – список кнопок для проведения действий над ними |
| Операции | Private void buttonAddNotion\_Click () – добавление понятия на сетку |
|  | Private void buttonSaveCros\_Click () – сохранение кроссворда в файл |
|  | Void GenCrossword () – генерация кроссворда на сетке |
|  | Private void GridButtonClicked () – регулирование правил заполнения сетки |
|  | Private void sortLen () – сортировка слова по выделенной длине на сетке |
|  | Private void sortLetter () – сортировка по алфавиту |

Таблица 3.7 – Реализация класса «Solution»

|  |  |
| --- | --- |
| Поля | Private int helpCount – количество подсказок |
| Private string [ , ] solutionGrid – решенная сетка кроссворда |
| Операции | Public Dictionary<string, string> GetDictionary () – получение словаря понятий |
|  | public Solution(Dictionary<string, string> \_dictionary, Grid \_grid, string[,] \_solutionGrid, int \_helpCount) – конструктор решения |

Таблица 3.8 – Реализация класса «Grid»

|  |  |
| --- | --- |
| Поля | Private int height – длина сетки |
| Private int width – ширина сетки |
| Операции | Public void AddWord () – добавление слова на сетку |
|  | Public void DeleteWord () – удаление слова с сетки |

Таблица 3.9 – Реализация класса «FormUser»

|  |  |
| --- | --- |
| Поля | Private RichTextBox[ , ] textBoxes – матрица текстовых полей, для вписывания букв |
| List<Word> words – список слов |
| Операции | Private void buttonChangedCrwd\_Click () – выбор словаря для разгадывания |
|  | Private void buttonHelp\_Click () – использование подсказок для разгадывания кроссворда |
|  | Private void buttonSaveSolution\_Click () – сохранение решения кроссворда |
|  | Private bool CheckGrid () – проверка правильности разгадывания |
|  | Private string FindKey () – поиск ключа словаря по значению |
|  | Private float ScaleFont () – динамическое изменение шрифта |

* 1. Реализация и описание модулей программы

Модульность − это свойство программы, связанное с декомпозицией ее на ряд ее отдельных фрагментов, которые компилируется по отдельности, но могут устанавливать связи между собой.

Диаграмма модулей описывает особенности физической реализации системы в момент перехода от логического представления к конкретной реализации системы. Она позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код

На рисунке 41 приведена диаграмма модулей. В таблице 3.10 приведено описание компонентов.

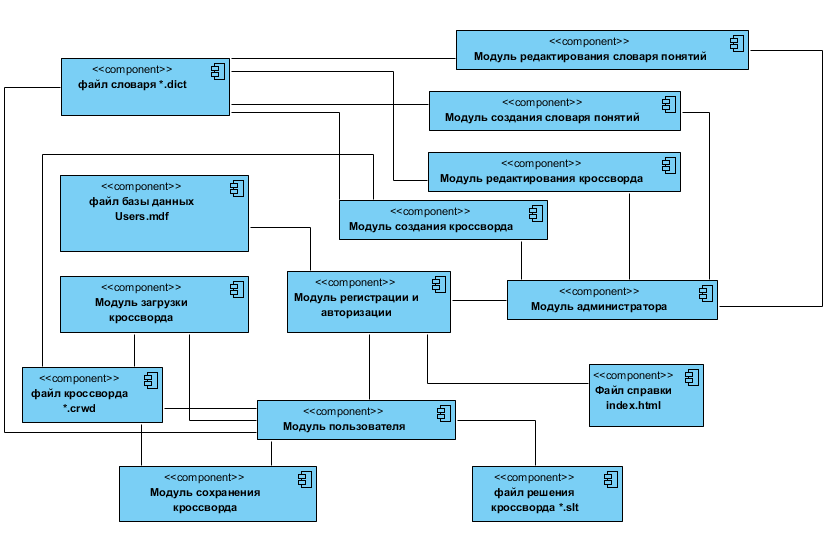


Рисунок 41 – Диаграмма компонентов

Таблица 3.10 – Таблица компонентов

|  |  |  |
| --- | --- | --- |
| Название компонента | Назначение компонента | Классы, входящие в компонент |
| 1 | 2 | 3 |
| Модуль создания кроссворда | Отвечает за создание кроссворда | FormBeforeCreateCros.cs;  Grid.cs  Word.cs |
| Модуль редактирования кроссворда | Отвечает за редактирование кроссворда | FormHandMadeCros.cs. |

Продолжение таблицы 3.10

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Модуль создания словаря понятий | Отвечает за создание словаря понятий | FormCreateDict.cs;  FormBeforeCreate; |
| Модуль редактирования словаря понятий | Отвечает за редактирование словаря понятий | FormAddNotion.cs; |
| Модуль администратора | Реализует функции администратора по работе с кроссвордом и словарем | FormAdmin.cs |
| Модуль пользователя | Отвечает за действия пользователя при работе с кроссвордом | FormUser.cs  Solution.cs |
| Модуль регистрации и авторизации | Отвечает за регистрацию и авторизацию пользователя в системе | Form1.cs  FormRegistration.cs |
| Модуль сохранения кроссворда | Отвечает за сохранение кроссворда | FormUser.cs  FormHandMadeCros.cs.  Файл кроссворда \*.crwd.  Файл решения кроссворда \*.slt. |
| Модуль загрузки кроссворда | Отвечает за загрузку кроссворда | Файл кроссворда \*.crwd.  Файл решения кроссворда \*.slt. |

.

* 1. Выбор и обоснование комплекса технических средств

Выбор комплекса технических средств в значительной степени определяет функциональные возможности и надежность создаваемой программы. Выбор комплекса технических средств производится на основе перечня решаемых задач и их информационных характеристик, спецификой обрабатываемой информации, техническими возможностями оборудования и т.д.

* + 1. Расчет объема занимаемой памяти

Расчет объема внешней памяти

Для того чтобы сделать и обосновать выбор комплекса технических средств, необходимо провести расчет объема занимаемой памяти (жесткого диска и ОЗУ).

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

VЖД = VОС + VПР+ VБД,

где VОС – объем памяти, занимаемый операционной системой (операционная система Windows 10, VОС = 20 Гб);

VПР – объем памяти, занимаемый непосредственно файлами приложения (VПР = 384 Кб);

VБД – объем памяти, занимаемый базой данных при ее максимальном заполнении. Расчет этой составляющей приведен в таблице 7;

VБД = 1400000 байт = 14 000 Кб = 1,367 Гб.

Таким образом, суммарный объем внешней памяти составит:

VЖД = 20 Гб +1,367Гб+384 Кб ≈ 21,5 Гб.

*Расчет объема ОЗУ*

Для расчета необходимого объема ОЗУ воспользуемся следующей формулой:

VОЗУ = VОС + VПР+VСПО,

Таблица 7 – Расчет объема внешней памяти, необходимой для хранения БД

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица | Размер записи (байт) | Максимум записей | Всего (байт) |
| Пользователь | 140 | 10000 | 1400000 |
| Итого | | | 1400000 |

где VОС – ОЗУ, занимаемое операционной системой (2 Гб);

VПР – ОЗУ, которое займет само приложение (не превысит 16 Мб);

VСПО – ОЗУ системного ПО, которое занимает приложение (не превысит 100 Мб)

Суммарные объемы ОЗУ составит:

VОЗУ = 2 Гб + 16 Мб+100МБ ≈ 2,2 Гб.

Таким образом, 4 Гб оперативной памяти можно счесть минимально необходимым для функционирования системы.

* + 1. Минимальные требования, предъявляемые к системе

Для корректного функционирования системы необходимо:

1. тип ЭВМ: x86-64 совместимый;
2. объем ОЗУ – не менее 4 Гб;
3. объем свободного дискового пространства – не менее 22 Гб;
4. клавиатура или иное устройство ввода;
5. монитор с разрешением 1024x768;
6. мышь или иное манипулирующее устройство.

ЗАКЛЮЧЕНИЕ

В рамках лабораторного практикума была разработана автоматизированная система для разгадывания классического кроссворда. Создание кроссворда совершается администратором в автоматическом и ручном режиме.

В первом разделе приведены результаты анализа предметной области, обзор аналогов системы, сформулирована постановка задачи.

На этапе проектирования системы были разработаны структурная схема и спецификация системы, продуман и проиллюстрирован прототип интерфейса пользователя и администратора и диаграмма классов, выбраны и приведены оптимальные алгоритмы функционирования системы, выбран и обоснован комплекс технических и программных средств.

На этапе реализации был разработан и описан интерфейс пользователя и администратора, приведена реализация сущностных классов, рассчитан объем занимаемой памяти и оперативной памяти и сформулированы требования, предъявляемые к системе.

Система, разработанная авторами, отлично подходит для людей, которые любят разгадывать классический кроссворд.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Кроссворд [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2017. URL: https://ru.wikipedia.org/wiki/  
   Кроссворд / (дата обращения: 29.09.2018).
2. Технология быстрой разработки RAD-приложений  [Электронный ресурс] // Студопедия: [сайт]. URL: https://studopedia.org/2-76756.html / (дата обращения: 03.10.2018).
3. Направления развития баз данных [Электронный ресурс]. URL: https://www.intuit.ru/studies/courses/508/364/lecture/8665 (дата обращения: 03.10.2018).
4. Современные технологии объектно-ориентированного анализа и проектирования информационных систем [Электронный ресурс] // Интуит – национальный открытый университет: [сайт]. URL: https://www.intuit.ru/studies/courses/32/32/lecture/1000?page=2 (дата обращения: 03.10.2018).
5. Что такое кроссворд - С.И. Ожегов, Н.Ю. Шведова Толковый словарь русского языка [Электронный ресурс] // Classes.ru: [сайт]. URL: https://classes.ru/all-russian/russian-dictionary-Ozhegov-term-13507.htm / (дата обращения: 14.10.2018).
6. Какие бывают разновидности кроссвордов [Электронный ресурс] // AIF.RU: [сайт]. URL: http://www.aif.ru/dontknows/eternal/  
   kakie\_byvayut\_raznovidnosti\_krossvordov (дата обращения: 14.10.2018).
7. Создать кроссворд и сканворд онлайн [Электронный ресурс] // Crosswordus.com: [сайт]. URL: http://crosswordus.com/ru/puzzlemaker (дата обращения: 14.10.2018).
8. Возможности программы HotPotatoes 6 [Электронный ресурс] // Мультиурок: [сайт]. URL: https://ru.wikipedia.org/wiki/  
   Пользовательский\_интерфейс (дата обращения: 17.11.2018).
9. Hotpatatoes Составление кроссворда [Электронный ресурс] // [Педагог.ru](https://vk.com/away.php?to=http%3A%2F%2F%CF%E5%E4%E0%E3%EE%E3.ru&cc_key=) : [сайт]. URL: [http://pedagogru.ru/hotpotatoes-составление-кроссворда](https://vk.com/away.php?to=http%3A%2F%2Fpedagogru.ru%2Fhotpotatoes-%F1%EE%F1%F2%E0%E2%EB%E5%ED%E8%E5-%EA%F0%EE%F1%F1%E2%EE%F0%E4%E0&cc_key=) / (дата обращения : 20.11.2018).
10. Объектно-ориентированная методика проектирования ис   
    [Электронный ресурс] /[Studfiles.net](https://vk.com/away.php?to=http%3A%2F%2FStudfiles.net&cc_key=): [сайт]. URL: [https://studfiles.net/  
    preview/423769/page:3](https://studfiles.net/preview/423769/page:3) /(дата обращения: 20.11.2018).
11. Определения понятия "система" [Электронный ресурс] // ФПИ Кубанского Государственного Аграрного Университета: [сайт]. URL: <http://fpi-kubagro.ru/opredeleniya-ponyatiya-sistema> / (дата обращения: 29.11.2018).
12. Структурная схема [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2017. URL: https://ru.wikipedia.org/  
    wiki/Структурная\_схема (дата обращения: 29.11.2017).
13. Спецификация качества программного средства [Электронный ресурс] // Студопедия: [сайт]. URL: http://studopedia.su/  
    2\_60245\_spetsifikatsiya-kachestva-programmnogo-sredstva.html (дата обращения: 01.12.2018).
14. Функциональная спецификация [Электронный ресурс] // Википедия: э-лектрон. энциклопедия. 2001-2017. URL: http://pandia.ru/text/78/247/74988.php (дата обращения: 05.12.2018).
15. Бабушкина И.А., Окулов С.М. Практикум по объектно-ориентированному программированию. Систем. Требования: Adobe Reader XI, экран 10". URL: http://files.lbz.ru/pdf/cC2542-9-ch.pdf (дата обращения 05.12.2018).
16. Качество программного обеспечения [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2017. URL: [https://ru.wikipedia.org/wiki/Качество\_программного\_обспечения /](https://ru.wikipedia.org/wiki/Качество_программного_обспечения%20/) (дата обращения: 05.12.2018).
17. Создание физической модели базы данных. Учет влияния транзакций [Электронный ресурс] // Интуит – национальный открытый университет: [сайт]. URL: https://www.intuit.ru/studies/  
    professional\_retraining/953/courses/191/lecture/4985?page=3 (дата обращения: 05.12.2018).
18. Разработка пользовательских интерфейсов [Электронный ресурс] // Pandia: [сайт]. URL: http://pandia.ru/text/78/247/74988.php (дата обращения: 05.12.2018).
19. Языки программирования [Электронный ресурс] // Lukmore: [сайт]. URL: <http://lurkmore.to/Языки_программирования> (дата обращения: 10.12.2018)
20. Microsoft Visual Studio [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2017. URL: [https://ru.wikipedia.org/  
    wiki/Microsoft\_Visual\_Studio](https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio). (дата обращения: 10.12.2018)
21. Понятие пользовательского интерфейса [Электронный ресурс] // Студопедия: [сайт]. URL: http://studopedia.ru/4\_153814\_udalennie-banki-dannih.html (дата обращения: 12.12.2018 г.).

ПРИЛОЖЕНИЕ А  
Руководство пользователя

А.1 Назначение системы

Система предназначена для составления и разгадывания кроссворда. В системе предусмотрено два режима: пользователь и администратор. В режиме пользователя имеется возможность загрузки из файла кроссворда и решения его. В режиме администратора доступны функции по созданию кроссворда в ручном и автоматическом режиме.

А.2 Условия работы системы

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

1) Требования к техническому обеспечению:

* тип ЭВМ: x86-64 совместимый;
* объем ОЗУ – не менее 4 Гб;
* объем свободного дискового пространства – не менее 22 Гб;
* клавиатура или иное устройство ввода;
* монитор с разрешением 1024x768;
* мышь или иное манипулирующее устройство.

2) Требования к программному обеспечению:

* операционная система Windows 8.1 и выше;

**А.3 Установка системы**

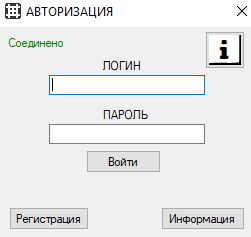
Система поставляется в виде rar-архива. Данный файл необходимо распаковать в любую директорию на жестком диске. Запускаемым файлом системы является файл Crossword.exe.

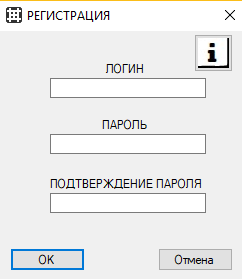
А.4 Работа с системой

А.4.1 Вход в систему

После запуска системы откроется окно авторизации, предназначенное для входа в систему. В данном окне пользователь может выполнить вход в систему или зарегистрироваться как новый пользователь, нажав на кнопку «Регистрация» (рисунок А.1).

Для регистрации нового пользователя необходимо ввести логин, пароль и подтверждение пароля (рисунок А.2).

  
Рисунок А.1 – Экранная форма авторизации

  
Рисунок А.2 – Экранная форма регистрации

А.4.2 Работа с системой в режиме администратора

Для того чтобы зайти в систему в режиме администратора необходимо ввести логин и пароль администратора в поля в окне авторизации (рисунок А.1), после чего будет осуществлен переход в главное меню (рисунок А.3). В этой форме можно создать кроссворд, редактировать уже созданный кроссворд, создать или отредактировать словарь понятий.

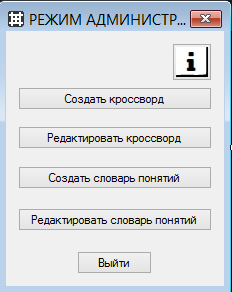


Рисунок А.3 – главное меню администратора

При нажатии на кнопку «Создать кроссворд» откроется форма, представленная на рисунке А.4, на которой отображаются параметры и средства для создания кроссворда.

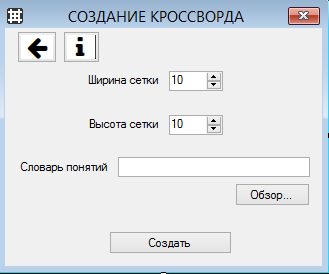


Рисунок А.4 – Экранная форма для создания кроссворда

После выбора параметров и при нажатии на кнопку «Создать» откроется форма «Создание/редактирование кроссворда» (рисунок А.5), на которой находятся элементы работы с кроссвордом. При нажатии на кнопку «редактировать кроссворд» в окне «Режим администратора» (рисунок А.3) также откроется окно «Создание/редактирование кроссворда» (рисунок А.5). Если в режиме администратора выбрать «Создать словарь понятий», то открывается экранная форма «Создание/редактирование словаря понятий» (рисунок А.6). В этой форме можно добавить новое понятие, нажав на кнопку с изображением «+» или выбрать понятие и нажать на кнопку с изображением карандаша. И в том и другом случае откроется форма

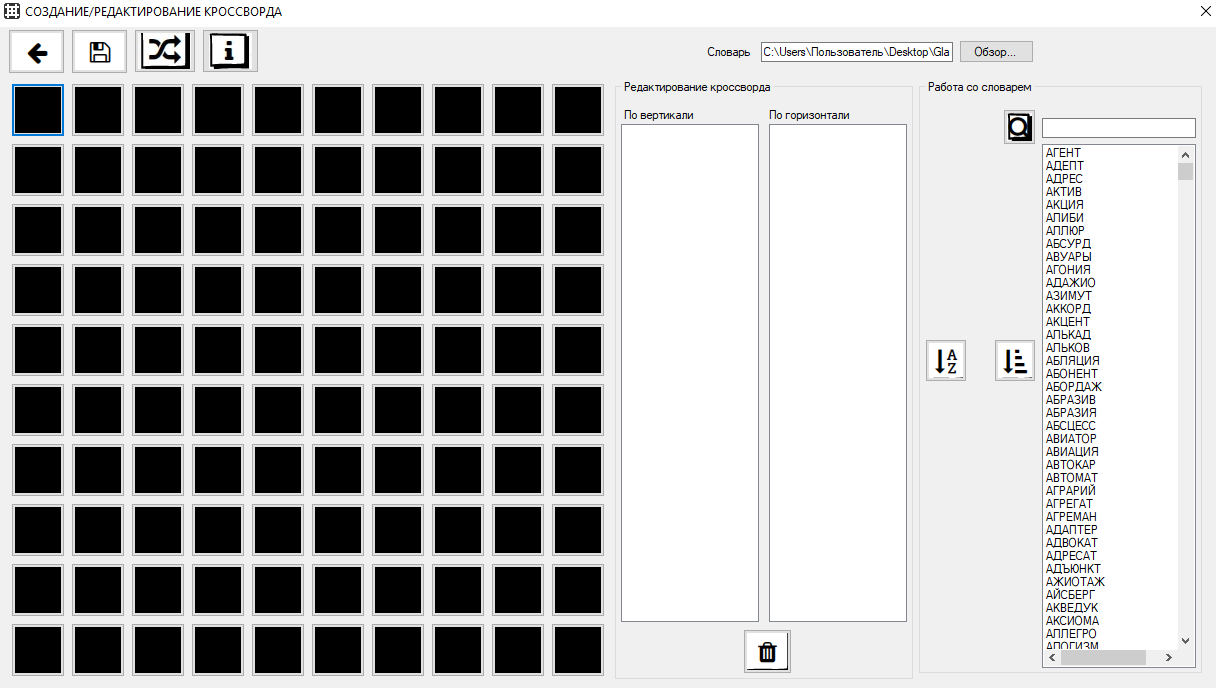


Рисунок А.5 – Экранная форма для создания и редактирования кроссворда

«Добавление/изменение понятия» (рисунок А.7). После добавления или изменения понятия необходимо нажать на кнопку «Сохранить», для сохранения понятия в словарь. При нажатии на кнопку «Редактировать словарь понятий» открывается окно «Создание/редактирование словаря понятий» (рисунок А.6).

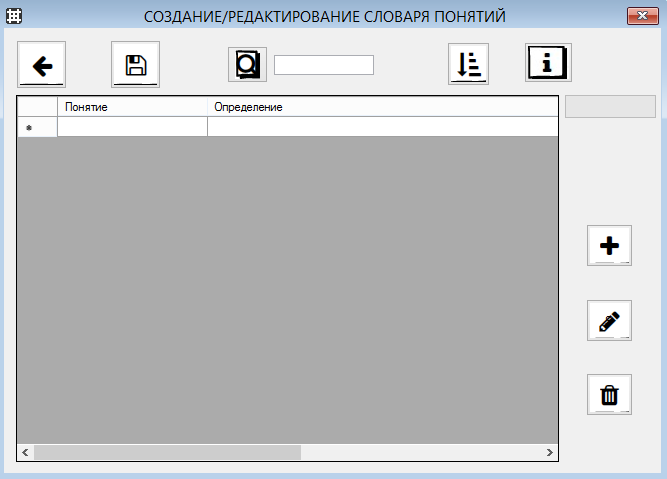


Рисунок А.6 – Экранная форма создания или редактирования словаря понятий

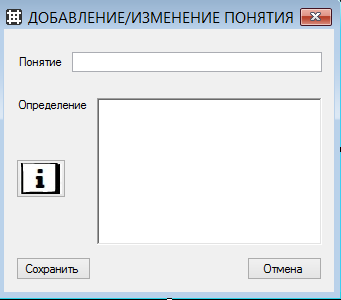


Рисунок А.7 – Экранная форма добавление/изменение понятия

**А.5 Системные ошибки**

***А.5.1 Справочная подсистема***

ПРИЛОЖЕНИЕ Б   
Листинг модулей программы

using Crossword.Autentification;

using Crossword.User;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Diagnostics;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Crossword

{

public partial class FormMain : Form

{

public FormMain()

{

InitializeComponent();

}

SqlConnection sqlConnection;

private void Form1\_Load(object sender, EventArgs e)

{

try

{

string connect = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\Пользователь\Documents\GitHub\Crossword\Crossword\UsersDB.mdf;Integrated Security=True";

sqlConnection = new SqlConnection(connect);

sqlConnection.Open();

labelConnect.ForeColor = Color.Green;

labelConnect.Text = "Соединено";

labelConnect.Visible = true;

buttonLogIn.Enabled = true;

buttonRegistration.Enabled = true;

}

catch (Exception)

{

labelConnect.ForeColor = Color.Red;

labelConnect.Text = "Соединение отсутствует";

labelConnect.Visible = true;

MessageBox.Show("Проблемы с соединением. Требуется перезапуск", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

}

private void button1\_Click(object sender, EventArgs e)

{

string login = textBoxLogin.Text;

string password = textBoxPas.Text;

if (!login.Equals("") && !password.Equals(""))

{

SqlCommand sqlCommand = new SqlCommand("SELECT\* FROM Users Where login ='" + login + "' " +

"and password ='" + password + "';", sqlConnection);

SqlDataReader reader = sqlCommand.ExecuteReader();

if (reader.Read())

{

bool isAdmin = (bool)reader["isAdmin"];

if (isAdmin)

{

FormAdmin formAdmin = new FormAdmin(this);

formAdmin.Show();

Visible = false;

}

else

{

CallOpenFileDialog();

}

textBoxLogin.Clear();

textBoxPas.Clear();

}

else

{

MessageBox.Show("Логин или пароль введен неверно", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

reader.Close();

}

}

private void CallOpenFileDialog()

{

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "Crossword |\*.crwd; \*.slt";

openFileDialog.Title = "Открыть кроссворд";

openFileDialog.ShowDialog();

bool isOpened = true;

if (openFileDialog.FileName != "")

{

FormUser formUser = new FormUser(this, openFileDialog.FileName, ref isOpened);

if (isOpened)

{

formUser.Show();

Visible = false;

}

else

{

CallOpenFileDialog();

}

}

}

private void buttonRegistration\_Click(object sender, EventArgs e)

{

FormRegistration registration = new FormRegistration(sqlConnection, this);

registration.Show();

Visible = false;

}

private void buttonInfo\_Click(object sender, EventArgs e)

{

MessageBox.Show("Лабораторный практикум по дисциплине\n" +

"'Технологии программирования'\n" +

"Классический кроссворд\n" +

"1. Байрамов Владимир Алексеевич\n " +

"2. Мавлютов Владимир Дмитриевич\n " +

"3. Перевозчиков Никита Дмитриевич \n " +

"4. Фёдоров Сергей Владимирович\n " +

"Самарский университет\n " +

"Группа 6402-090301D\n" +

"Все права защищены.", "АВТОРЫ");

}

private void textBoxPas\_KeyPress(object sender, KeyPressEventArgs e)

{

char l = e.KeyChar;

if ((l < '0' || l > '9') && (l < 'A' || l > 'z') && l != '\b' && l != '.')

{

e.Handled = true;

MessageBox.Show("Ввод не латиницы и не цифр запрещен");

}

}

private void textBoxLogin\_KeyPress(object sender, KeyPressEventArgs e)

{

char l = e.KeyChar;

if ((l < '0' || l > '9') && (l < 'A' || l > 'z') && l != '\b' && l != '.')

{

e.Handled = true;

MessageBox.Show("Ввод не латиницы и не цифр запрещен");

}

}

private void button1\_Click\_1(object sender, EventArgs e)

{

try

{

Process.Start(@"D:\Repository\Crossword\index.html");

}

catch (System.ComponentModel.Win32Exception)

{

MessageBox.Show("Файл справки не найден", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

}

}

}

using Crossword.User;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Diagnostics;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Crossword.Autentification

{

public partial class FormRegistration : Form

{

SqlConnection sqlConnection;

FormMain formMain;

public FormRegistration(SqlConnection sqlConnection, FormMain formMain)

{

this.sqlConnection = sqlConnection;

this.formMain = formMain;

InitializeComponent();

}

private void FormRegistration\_Load(object sender, EventArgs e)

{

}

private void buttonCancel\_Click(object sender, EventArgs e)

{

Close();

}

private void buttonOk\_Click(object sender, EventArgs e)

{

string login = textBoxLogin.Text;

string pass = textBoxPassword.Text;

string passY = textBoxPasswordYet.Text;

if (!login.Equals("") && !pass.Equals("") && !passY.Equals(""))

{

if (login.Length > 5 && pass.Length > 5)

{

if (pass.Equals(passY))

{

SqlCommand sqlCommand = new SqlCommand("Select\* from Users Where login = '" + login + "';",

sqlConnection);

SqlDataReader reader = sqlCommand.ExecuteReader();

if (!reader.Read())

{

reader.Close();

sqlCommand = new SqlCommand("Insert into Users (login, password, isAdmin) " +

"values ('" + login + "', '" + pass +

"', 0);", sqlConnection);

sqlCommand.ExecuteNonQuery();

MessageBox.Show("Успешно");

Close();

}

else

{

MessageBox.Show("Такой пользователь уже есть", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

reader.Close();

}

else

{

MessageBox.Show("Пароли не совпадают", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error,

MessageBoxDefaultButton.Button1);

}

}

else

{

MessageBox.Show("Слишком короткий логин или пароль", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error,

MessageBoxDefaultButton.Button1);

}

}

}

private void FormRegistration\_FormClosing(object sender, FormClosingEventArgs e)

{

formMain.Visible = true;

}

private void textBoxPassword\_KeyPress(object sender, KeyPressEventArgs e)

{

char l = e.KeyChar;

if ((l < '0' || l > '9') && (l < 'A' || l > 'z') && l != '\b' && l != '.')

{

e.Handled = true;

MessageBox.Show("Ввод не латиницы и не цифр запрещен");

}

//else if ((l < 'А' || l > 'я') && l != '\b' && l != '.') { MessageBox.Show("Ввод не латиницы и цифр запрещен"); }

}

private void textBoxPasswordYet\_KeyPress(object sender, KeyPressEventArgs e)

{

char l = e.KeyChar;

if ((l < '0' || l > '9') && (l < 'A' || l > 'z') && l != '\b' && l != '.')

{

e.Handled = true;

MessageBox.Show("Ввод не латиницы и не цифр запрещен");

}

}

private void textBoxLogin\_KeyPress(object sender, KeyPressEventArgs e)

{

char l = e.KeyChar;

if ((l < '0' || l > '9') && (l < 'A' || l > 'z') && l != '\b' && l != '.')

{

e.Handled = true;

MessageBox.Show("Ввод не латиницы и не цифр запрещен");

}

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

Process.Start(@"C:\Users\nikit\Documents\GitHub\Crossword\index.html");

}

catch (System.ComponentModel.Win32Exception)

{

MessageBox.Show("Файл справки не найден", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

}

}

}

using Crossword.Admin;

using Crossword.Admin.CreateEditDict;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Diagnostics;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Crossword

{

public partial class FormAdmin : Form

{

private FormMain formMain;

public FormAdmin(FormMain formMain)

{

this.formMain = formMain;

InitializeComponent();

}

private void buttonClose\_Click(object sender, EventArgs e)

{

Close();

}

private void buttonOpenCreateCros\_Click(object sender, EventArgs e)

{

FormBeforeCreate formBefore = new FormBeforeCreate(this);

formBefore.Show();

Visible = false;

}

private void buttonOpenCreateDict\_Click(object sender, EventArgs e)

{

FormCreateDict formDict = new FormCreateDict(this);

formDict.Show();

Visible = false;

}

private void buttonOpenEditDict\_Click(object sender, EventArgs e)

{

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "Text |\*.txt";

openFileDialog.Title = "Открыть словарь";

openFileDialog.ShowDialog();

if (openFileDialog.FileName != "")

{

FormCreateDict formDict = new FormCreateDict(this, openFileDialog.FileName);

formDict.Show();

Visible = false;

}

}

private void buttonOpenEditCros\_Click(object sender, EventArgs e)

{

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "CrosswordFile |\*.crwd";

openFileDialog.Title = "Открыть кроссворд";

openFileDialog.ShowDialog();

if (openFileDialog.FileName != "")

{

bool check = true;

FormHandMadeCros formDict = new FormHandMadeCros(this, openFileDialog.FileName, ref check);

if (check)

{

formDict.Show();

Visible = false;

}

}

}

private void FormAdmin\_FormClosing(object sender, FormClosingEventArgs e)

{

formMain.Visible = true;

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

Process.Start(@"D:\Repository\Crossword\index.html");

}

catch (System.ComponentModel.Win32Exception)

{

MessageBox.Show("Файл справки не найден", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

}

private void FormAdmin\_Load(object sender, EventArgs e)

{

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Diagnostics;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Crossword.Admin.CreateEditDict

{

public partial class FormCreateDict : Form

{

private FormAdmin formAdmin;

private Dictionary<string, string> dictionary;

private string fileName;

public FormCreateDict(FormAdmin formAdmin)

{

dictionary = new Dictionary<string, string>();

this.formAdmin = formAdmin;

InitializeComponent();

}

public FormCreateDict(FormAdmin formAdmin, string fileName)

{

dictionary = new Dictionary<string, string>();

this.fileName = fileName;

this.formAdmin = formAdmin;

StreamReader streamReader = new StreamReader(fileName, Encoding.GetEncoding("Windows-1251"));

InitializeComponent();

try

{

string dataFromFile = "";

while (dataFromFile != null)

{

dataFromFile = streamReader.ReadLine();

if (dataFromFile != null)

{

if (!dataFromFile.Equals(""))

{

string[] stringArr = dataFromFile.Split(' ');

string notion = stringArr[0];

string def = "";

for (int i = 1; i < stringArr.Length - 1; i++)

{

def += stringArr[i] + " ";

}

def += stringArr[stringArr.Length - 1];

dictionary.Add(notion.ToUpper(), def.ToLower());

}

}

}

}

catch (ArgumentException)

{

MessageBox.Show("Понятие повторяется!", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

streamReader.Close();

}

}

private void updateGrid()

{

if (dictionary.Count != 0)

{

dataGridView.Rows.Clear();

progressBar1.Value = 0;

progressBar1.Minimum = 0;

progressBar1.Maximum = dictionary.Count;

progressBar1.Step = 1;

for (int i = 0; i < dictionary.Count; i++)

{

dataGridView.Rows.Add();

dataGridView.Rows[i].Cells[0].Value = dictionary.ElementAt(i).Key;

dataGridView.Rows[i].Cells[1].Value = dictionary.ElementAt(i).Value;

progressBar1.PerformStep();

}

}

}

private void buttonBack\_Click(object sender, EventArgs e)

{

Close();

}

private void FormCreateDict\_Load(object sender, EventArgs e)

{

}

private FormAddNotion formAdd;

private void buttonAdd\_Click(object sender, EventArgs e)

{

formAdd = new FormAddNotion();

formAdd.Show();

formAdd.buttonSave.Click += new EventHandler(buttonSaveNotion\_Click);

formAdd.FormClosing += FormAddEditNotion\_Closing;

Enabled = false;

}

private void FormAddEditNotion\_Closing(object sender, FormClosingEventArgs e)

{

buttonAdd.Enabled = true;

buttonEdit.Enabled = true;

buttonDelete.Enabled = true;

Enabled = true;

}

private void buttonSaveNotion\_Click(object sender, EventArgs e)

{

try

{

string notion = formAdd.textBoxNotion.Text.ToUpper();

string def = formAdd.richTextBoxDef.Text.ToLower();

if (notion.Equals("") || def.Equals(""))

{

MessageBox.Show("Поля не должны быть пустыми!", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

else

{

dictionary.Add(notion, def);

dataGridView.Rows.Add();

dataGridView.Rows[dataGridView.Rows.Count - 2].Cells[0].Value = notion;

dataGridView.Rows[dataGridView.Rows.Count - 2].Cells[1].Value = def;

formAdd.Close();

}

}

catch (System.ArgumentException)

{

MessageBox.Show("Понятие повторяется!", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

}

private string oldNot;

private void buttonEdit\_Click(object sender, EventArgs e)

{

if (dataGridView.CurrentRow.Cells[0].Value != null)

{

string not = dataGridView.CurrentRow.Cells[0].Value.ToString();

string def = dataGridView.CurrentRow.Cells[1].Value.ToString();

oldNot = not;

formAdd = new FormAddNotion(not, def);

formAdd.buttonSave.Click += new EventHandler(buttonSaveEditedNotion\_Click);

formAdd.Show();

Enabled = false;

}

}

private void buttonSaveEditedNotion\_Click(object sender, EventArgs e)

{

try

{

string not = formAdd.textBoxNotion.Text.ToUpper();

string def = formAdd.richTextBoxDef.Text.ToLower();

if (not.Equals("") || def.Equals(""))

{

MessageBox.Show("Поля не должны быть пустыми!", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

else

{

if (oldNot.Equals(not))

{

dictionary[not] = def;

dataGridView.CurrentRow.Cells[1].Value = def;

formAdd.Close();

}

else

{

dictionary.Remove(oldNot);

dictionary.Add(not, def);

dataGridView.CurrentRow.Cells[0].Value = not;

dataGridView.CurrentRow.Cells[1].Value = def;

dataGridView.Rows.Add();

formAdd.Close();

}

}

}

catch (ArgumentException)

{

MessageBox.Show("Понятие повторяется!", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

}

private void buttonDelete\_Click(object sender, EventArgs e)

{

if (dataGridView.CurrentRow.Cells[0].Value != null)

{

string not = dataGridView.CurrentRow.Cells[0].Value.ToString();

string def = dataGridView.CurrentRow.Cells[1].Value.ToString();

dictionary.Remove(not);

dataGridView.Rows.Remove(dataGridView.CurrentRow);

}

}

private bool isSortLen;

private void sortLen()

{

List<string> strings = new List<string>();

for (int i = 0; i < dictionary.Count; i++)

{

strings.Add(dictionary.ElementAt(i).Key);

}

if (!isSortLen)

{

strings.Sort(delegate (string s1, string s2) { return s1.Length.CompareTo(s2.Length); });

isSortLen = true;

}

else

{

strings.Sort(delegate (string s1, string s2) { return -s1.Length.CompareTo(s2.Length); });

isSortLen = false;

}

dataGridView.Rows.Clear();

for (int i = 0; i < strings.Count; i++)

{

dataGridView.Rows.Add();

dataGridView.Rows[i].Cells[0].Value = strings[i];

dataGridView.Rows[i].Cells[1].Value = dictionary[strings[i]];

progressBar1.PerformStep();

}

}

private void buttonSortLen\_Click(object sender, EventArgs e)

{

Enabled = false;

progressBar1.Value = 0;

progressBar1.Minimum = 0;

progressBar1.Maximum = dictionary.Count;

progressBar1.Step = 1;

sortLen();

Enabled = true;

}

private void buttonSaveDict\_Click(object sender, EventArgs e)

{

Enabled = false;

SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.Filter = "Text |\*.txt";

saveFileDialog.Title = "Сохранить словарь";

saveFileDialog.ShowDialog();

if (saveFileDialog.FileName != "")

{

Stream s = new FileStream(saveFileDialog.FileName, FileMode.Create);

StreamWriter stream = new StreamWriter(s, Encoding.GetEncoding("Windows-1251"));

progressBar1.Value = 0;

progressBar1.Maximum = dictionary.Count;

for (int i = 0; i < dictionary.Count; i++)

{

stream.WriteLine(dictionary.ElementAt(i).Key + " " + dictionary.ElementAt(i).Value);

progressBar1.PerformStep();

}

stream.Close();

}

Enabled = true;

}

private void button1\_Click(object sender, EventArgs e)

{

if (!textBoxSearch.Text.Equals(""))

{

Enabled = false;

dataGridView.Rows.Clear();

progressBar1.Value = 0;

progressBar1.Minimum = 0;

progressBar1.Maximum = dictionary.Count;

progressBar1.Step = 1;

for (int i = 0, j = 0; i < dictionary.Count; i++)

{

string key = dictionary.ElementAt(i).Key;

if (key.StartsWith(textBoxSearch.Text.ToUpper()))

{

dataGridView.Rows.Add();

dataGridView.Rows[j].Cells[0].Value = key;

dataGridView.Rows[j].Cells[1].Value = dictionary.ElementAt(i).Value;

j++;

}

progressBar1.PerformStep();

}

Enabled = true;

}

else

{

updateGrid();

}

}

private void FormCreateDict\_FormClosing(object sender, FormClosingEventArgs e)

{

formAdmin.Visible = true;

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

Process.Start(@"D:\Repository\Crossword\index.html");

}

catch (System.ComponentModel.Win32Exception)

{

MessageBox.Show("Файл справки не найден", "Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error, MessageBoxDefaultButton.Button1);

}

}

private void FormCreateDict\_Shown(object sender, EventArgs e)

{

updateGrid();

}

}

}

1. *Количество страниц, рисунков, таблиц указывается с учетом приложений* [↑](#footnote-ref-1)