МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

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

Институт информатики и кибернетики

Кафедра программных систем

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
  
 к курсовой работе по дисциплине «Программная инженерия»

по теме «Автоматизированная система генерирования

структуры лабиринта и нахождения выхода из него»

Обучающийся Е.А. Балашова

Обучающийся В.А. Гриднева

Руководитель Л.С. Зеленко

Самара 2024

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

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

Институт информатики и кибернетики

Кафедра программных систем

ЗАДАНИЕ

на курсовую работу по дисциплине

«Программная инженерия»

обучающимся в группе № 6402-020302D

Е.А. Балашовой

В.А. Гридневой

Тема проекта:«Автоматизированная система генерирования структуры

лабиринта и нахождения выхода из него»

1. Исходные данные к проекту**:** см. приложение к заданию
2. Перечень вопросов, подлежащих разработке:
   1. Произвести анализ предметной области: изучить основные принципы составления лабиринтов, изучить алгоритмы генерации лабиринтов и их решения
   2. Выполнить обзор существующих систем-аналогов
   3. Разработать информационно-логический проект системы по методологии UML
   4. Разработать и реализовать программное и информационное обеспечение, провести его тестирование и отладку .
   5. Оформить документацию курсовой работы
   6. Подготовить презентацию по разработанной системе
3. Перечень графических разработок:
   1. Структурная схема системы
   2. Канонические диаграммы UML
   3. Схемы основных алгоритмов
4. 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 | 20.09.2024 |  |
| 2 | Описание и анализ предметной области | 10 | 27.09.2024 |  |
| 3 | Проектирование системы | 40 | 13.12.2024 |  |
| 3.1 | Разработка структурной схемы системы | 5 | 11.10.2024 |  |
| 3.2 | Разработка функциональной спецификации системы и прототипа интерфейса пользователя | 10 | 25.10.2024 |  |
| 3.3 | Разработка информационно-логического проекта системы и его предъявление руководителю | 25 | 13.12.2024 |  |
| 4 | Реализация проекта, разработка контрольных примеров. Предъявление реализации руководителю | 40 | 13.12.2024 |  |
| 5 | Корректировка проекта и оформление документации проекта. Защита проекта с представлением презентации. | 5 | 27.12.2024 |  |

Задание принял  
 к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.А. Балашова

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.А. Гриднева

ПРИЛОЖЕНИЕ  
к заданию на курсовую работу  
обучающимся в группе № 6402-020302D  
Е.А. Балашовой

В.А. Гридневой

Тема проекта: «Автоматизированная система генерирования структуры лабиринта и нахождения выхода из него»

Исходные данные к проекту:

1. Характеристика объекта автоматизации:

## объект автоматизации: лабиринт;

## виды автоматизируемой деятельности:

* + процесс авторизации и регистрации пользователей;
  + процесс генерирования лабиринта в соответствии с алгоритмами;
  + процесс расстановки входа и выхода;
  + процесс автоматического прохождения лабиринта;
  + процесс визуализации нахождения выхода из лабиринта;

## количество ролей пользователей – 2;

## минимальный размер лабиринта по вертикали – 7;

## максимальный размер лабиринта по вертикали – 21;

## минимальный размер лабиринта по горизонтали – 7;

## максимальный размер лабиринта по горизонтали – 21;

## количество входов – 1;

## количество выходов – 1;

## количество способов расстановки входа и выхода – 2;

## количество алгоритмов создания лабиринта – 2;

## количество тем оформления лабиринта – 4;

## количество алгоритмов прохождения лабиринта – 2;

## количество скоростных режимов перемещения персонажа – 4;

## минимальная длина логина – 6 символов;

## максимальная длина логина – 16 символов;

## минимальная длина пароля – 6 символов;

## максимальная длина пароля – 20 символов;

1. Требования к информационному обеспечению:
2. информационное обеспечение разрабатывается на основе следующих источников:
   * лабиринты: классификация, генерирование, поиск решений [Электронный ресурс]. URL: https://habr.com/ru/articles/445378/ (дата обращения: 16.09.2024);
   * алгоритмы генерации лабиринтов и нахождения пути [Электронный ресурс]. URL: https://tproger.ru/articles/maze-generators (дата обращения: 16.09.2024);
   * генерация лабиринтов: алгоритм Эллера [Электронный ресурс]. URL: https://habr.com/ru/articles/667576/ (дата обращения: 16.09.2024);
3. требования к построению лабиринта:
   * лабиринт имеет формат 2D;
   * по периметру лабиринта – стена;
   * толщина стен равна толщине прохода;
   * вход и выход находится на периметре лабиринта, но не в углах;
   * вход и выход не совпадают;
   * стены не должны образовывать изолированные части;
   * наличие тупиков;
   * наличие пути;
4. лабиринты хранятся в файлах, структура файла определяется в процессе проектирования.
5. Требования к техническому обеспечению:
   1. Требования к техническому обеспечению серверной части:
6. тип ЭВМ – IBM PC совместимый;
7. объем ОЗУ – не менее 2 Гб;
8. объем свободного пространства на внешнем диске – не менее 50 Гб;
9. наличие подключения к сети Интернет;
10. манипулятор – мышь;
11. технические характеристики определяются в процессе выполнения проекта;
    1. Требования к техническому обеспечению клиентской части:
12. тип ЭВМ – IBM PC совместимый;
13. монитор с разрешающей способностью не ниже 800 х 600;
14. манипулятор – мышь;
15. технические характеристики определяются в процессе выполнения проекта.
16. Требования к программному обеспечению:
    1. Требования к программному обеспечению серверной части:
17. тип операционной системы – Windows 10 и выше.
    1. Требования к программному обеспечению клиентской части:
18. тип операционной системы – Windows 10 и выше;
19. браузер – Google Chrome 86.0.4240.183 (64-битный) и выше.
    1. Требования к программному обеспечению рабочего места разработчика:
20. тип операционной системы – Windows 10 и выше;
21. язык программирования – С#;
22. среда программирования – Visual Studio 2022;
23. среда проектирования – StarUML 5.2.0.
24. Общие требования к проектируемой системе:

5.1 Функции, реализуемые системой:

1. функции системы:
   * аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
   * генерирование шаблона лабиринта по заданному размеру;
   * случайная расстановка входа и выхода;
   * контроль расстановки входа и выхода;
   * создание структуры лабиринта по заданному алгоритму;
   * проверка структуры лабиринта при загрузке;
   * визуализация генерации лабиринта;
   * визуализация прохождения лабиринта (видимая дорожка);
2. функции администратора:
   * авторизация пользователя в системе (ввод логина и пароля);
   * настройка параметров лабиринта при создании:
3. задание размера по горизонтали;
4. задание размера по вертикали;
5. выбор способа расстановки входа и выхода;
6. выбор алгоритма генерации лабиринта;
7. настройки темы оформления;
   * ручная расстановка входа и выхода;
   * сохранение лабиринта в файл заданной структуры;
   * просмотр лабиринта;
8. функции игрока:
   * регистрация пользователя в системе (ввод логина и пароля);
   * авторизация пользователя в системе (ввод логина и пароля);
   * выбор лабиринта;
   * загрузка лабиринта из файла;
   * изменение темы оформления лабиринта;
   * выбор алгоритма прохождения лабиринта;
   * ручное прохождение лабиринта[[1]](#footnote-1);
   * выбор алгоритма прохождения лабиринта;
   * выбор режима прохождения лабиринта;
   * выбор задержки перемещения персонажа (для автоматического режима);
   * запуск алгоритма.

5.2 Технические требования к системе:

1. режим работы – диалоговый;
2. время автоматической генерации лабиринта – не более 5 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 13.09.2024 В.А. Гриднева

13.09.2024 Е.А. Балашова

РЕФЕРАТ

Пояснительная записка 40 с, 14 рисунков, 5 таблиц[[2]](#footnote-2), 12 источников,  
2 приложения.

Графическая часть: ??? слайдов презентации PowerPoint.

ДЕРЕВО ПОИСКА, ГЕНЕРАТОР ЛАБИРИНТОВ, ГОЛОВОЛОМКА, СЛОВАРЬ ТЕРМИНОВ, ВАРИАНТ ОТОБРАЖЕНИЯ, РАЗГАДЫВАНИЕ

Объектом автоматизации является линейный кроссворд.

Во время курсового проектирования разработаны алгоритмы и соответствующая им программа, позволяющая выполнять автоматическую генерацию линейного кроссворда по заданной теме. Задания (понятие и его расшифровка) хранятся в текстовом файле и могут дополняться вручную 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 2015 и функционирует под управлением операционной системы Windows 7 и выше.

СОДЕРЖАНИЕ

[Введение 12](#_Toc90897433)

[1 Описание и анализ предметной области 14](#_Toc90897434)

[1.1 Описание предметной области 14](#_Toc90897435)

[1.2 Описание систем-аналогов 15](#_Toc90897436)

[1.2.1 Название системы-аналога 1 15](#_Toc90897437)

[1.2.2 Название системы-аналога 2 16](#_Toc90897438)

[1.3 Диаграмма объектов предметной области 20](#_Toc90897439)

[1.4 Постановка задачи 20](#_Toc90897440)

[2 Проектирование системы 22](#_Toc90897441)

[2.1 Выбор и обоснование архитектуры системы 22](#_Toc90897442)

[2.2 Структурная схема системы 22](#_Toc90897443)

[2.3 Разработка спецификации требований 26](#_Toc90897444)

[2.3.1 Функциональная спецификация 26](#_Toc90897445)

[2.3.2 Перечень исключительных ситуаций 26](#_Toc90897446)

[2.4 Разработка прототипа интерфейса пользователя системы 29](#_Toc90897447)

[2.5 Разработка информационно-логического проекта системы 30](#_Toc90897448)

[2.5.3 Язык UML 32](#_Toc90897449)

[2.5.4 Диаграмма вариантов использования 32](#_Toc90897450)

[2.5.5 Сценарии 32](#_Toc90897451)

[2.5.6 Диаграмма классов 34](#_Toc90897452)

[2.5.7 Диаграмма состояний 34](#_Toc90897453)

[2.5.8 Диаграмма деятельности 34](#_Toc90897454)

[2.5.9 Диаграмма последовательности 35](#_Toc90897455)

[2.6 Логическая модель данных (при необходимости) 36](#_Toc90897456)

[2.7 Выбор и обоснование алгоритмов обработки данных /Разработка и описание алгоритмов обработки данных 37](#_Toc90897457)

[2.8 Выбор и обоснование комплекса программных средств 39](#_Toc90897458)

[2.8.1 Выбор языка программирования 39](#_Toc90897459)

[2.8.2 Выбор операционной системы 40](#_Toc90897460)

[2.8.3 Выбор среды программирования 40](#_Toc90897461)

[2.8.4 Выбор системы управления базами данных (при необходимости) 40](#_Toc90897462)

[3 Реализация системы 41](#_Toc90897463)

[3.1 Разработка и описание интерфейса пользователя 41](#_Toc90897464)

[3.2 Диаграммы реализации 42](#_Toc90897465)

[3.2.1 Диаграмма компонентов 42](#_Toc90897466)

[3.2.2 Диаграмма развертывания 43](#_Toc90897467)

[3.2.3 Диаграмма классов 43](#_Toc90897468)

[3.3 Физическая модель данных (при необходимости) 44](#_Toc90897469)

[3.4 Выбор и обоснование комплекса технических средств 45](#_Toc90897470)

[3.4.1 Расчет объема занимаемой памяти 45](#_Toc90897471)

[3.4.2 Минимальные требования, предъявляемые к системе 47](#_Toc90897472)

[Заключение 48](#_Toc90897473)

[Список использованных источников 49](#_Toc90897474)

[Приложение А Руководство пользователя 53](#_Toc90897475)

[А.1 Назначение системы 53](#_Toc90897476)

[А.2 Условия работы системы 53](#_Toc90897477)

[А.3 Установка системы 53](#_Toc90897478)

[А.4 Работа с системой 54](#_Toc90897479)

[А.4.1 Работа с системой в режиме администратора (если необходимо) 54](#_Toc90897480)

[А.4.2 Работа с системой в режиме пользователя 54](#_Toc90897481)

[Приложение Б Листинг модулей программы 55](#_Toc90897482)

ВВЕДЕНИЕ

Слово «лабиринт» пришло в русский язык из немецкого и является его прямым переводом. Первоисточник этого слова – греческий язык, в котором под лабиринтом понималось обширное пространство, состоящее из многочисленных комнат, коридоров и переходов, расположенных по запутанному плану, составленному с целью запутать незнающего человека. Другим значением этого слова считалось подземелье – природные подземные ходы, в которых можно было легко запутаться и потеряться [1].

Лабиринты являются частью культуры разных стран и эпох. Например, при разговоре об античной эпохе слово «лабиринт» ассоциируется с островом Крит и Минотавром. В Древнем Египте лабиринты строились в центре городов и использовались фараонами для управления страной и в религиозных целях. В Индии и Китае считалось, что лабиринты защищают от злых духов, поэтому входы в дома и города обязательно строили в виде лабиринтов.

В современном мире лабиринты имеют более развлекательных характер. Их используют в декоративных целях в ландшафтном дизайне, для тренировки мозга в компьютерных и настольных играх. Исследования показывают, что прохождение лабиринтов учит детей ориентироваться в пространстве и развивает память [2].

Во многих компьютерных играх встречаются лабиринты. Они могут быть как отдельным уровнем в логических играх, так и частью мира в приключенческих. Получается, что пользователя компьютера в работе с лабиринтами ожидают две основные задачи: генерация лабиринта и его прохождение.

Во время курсового проектирования необходимо разработать автоматизированную систему, с помощью которой можно генерировать лабиринт в автоматическом режиме с помощью специальных алгоритмов, а также находить выход из лабиринта вручную или используя алгоритмы нахождения пути.

Разработка системы будет производиться по технологии быстрой разработки приложений RAD (Rapid Application Development)*,* которая поддерживается методологией структурного проектирования и включает элементы объектно-ориентированного проектирования и анализа предметной области [3].

При проектировании системы будут использоваться методология ООАП (Object-Oriented Analysis/Design), в основу которой положена объектно-ориентированная методология представления предметной области в виде объектов, являющихся экземплярами соответствующих классов, и язык моделирования UML (Unified Modeling Language), который является стандартным инструментом для разработки «чертежей» программного обеспечения [4,5].

1. Описание и анализ предметной области

"Лабиринт" переводится как ущелье, подземная пещера с огромным количеством ходов, перекрестков и тупиков. Но существуют лабиринты и искусственного происхождения, созданные человеком вольно или невольно. Примером невольного создания лабиринтов являются различные шахты, рудники, каменоломни, называемые общим словом "катакомбы".

Чаще всего под словом "лабиринт" мы подразумеваем искусственное, специально созданное, сложное сооружение. Происхождение таких "лабиринтов" довольно древнее. Первоначально лабиринты носили наказательно-профилактический характер, но в дальнейшем стали предметом развлечений, перейдя в сады и парки в виде живой изгороди сложной конфигурации. В наше время, по утверждению Мартина Гарднера, "существует две области науки, в которых интерес к Лабиринтам остается неизменно высоким: психология и конструирование вычислительных машин.

На сегодняшний день применение алгоритмов построения и прохождения лабиринтов улучшает оптимизацию и логику построения различных математических объектов. В технических отраслях применение лабиринтов также широко используется на практике. Они используются для трассировки печатных плат, при создании динамических маршрутов и путей движения игроков, в телекоммуникациях для нахождения самых быстрых и самых коротких путей прохождения сетей, в горнодобывающей промышленности для оптимального построения шахт и др. Применение лабиринтов бывает и прикладное, например для обучения систем искусственного интеллекта.

* 1. Описание предметной области
     1. Описание лабиринта

Лабиринт – структура, состоящая из запутанных путей, ведущих к выходу или в тупик.

С математической точки зрения лабиринт представляет собой топологическую задачу. Если у лабиринта имеется только один вход и необходимо найти дорогу к единственному выходу, то такую задачу всегда можно решить. Для этого достаточно, идя по лабиринту, все время одной рукой касаться стенки. Таким образом можно всегда найти выход из лабиринта, но путь не будет кратчайшим.

Тот же метод пригоден и в более традиционном случае, когда цель находится внутри лабиринта. Но только если в нем нет путей, по которым можно кружить вокруг цели и возвращаться в исходную точку. Иначе попасть внутрь "островка", вокруг которого проходит замкнутый маршрут, не удастся.

* + 1. Классификация лабиринтов

Лабиринты можно разбить по четырем различным классификациям: размерность, происхождение, топология, тесселяция и маршрутизация [6].

* + - 1. Размерность определяет количество измерений, в которых лабиринт находится.

Двумерный лабиринт расположен в двух измерениях и может быть легко отображен на бумаге, на рисунке 1 приведен пример такого лабиринта.

  
Рисунок – Двумерный лабиринт

Трехмерный лабиринт имеет несколько уровней, проходы могут подниматься вверх и опускаться вниз. Пример трехмерного лабиринта приведен на рисунке 2.

  
Рисунок – Трехмерный лабиринт

Более высокие размерности лабиринтов трудно смоделировать в реальном мире, но они могут быть представлены математической абстракцией. Например, четырехмерный лабиринт может иметь порталы во времени, что будет интересным элементом компьютерной игры.

* + - 1. По происхождению лабиринты могут быть естественными и искусственными.

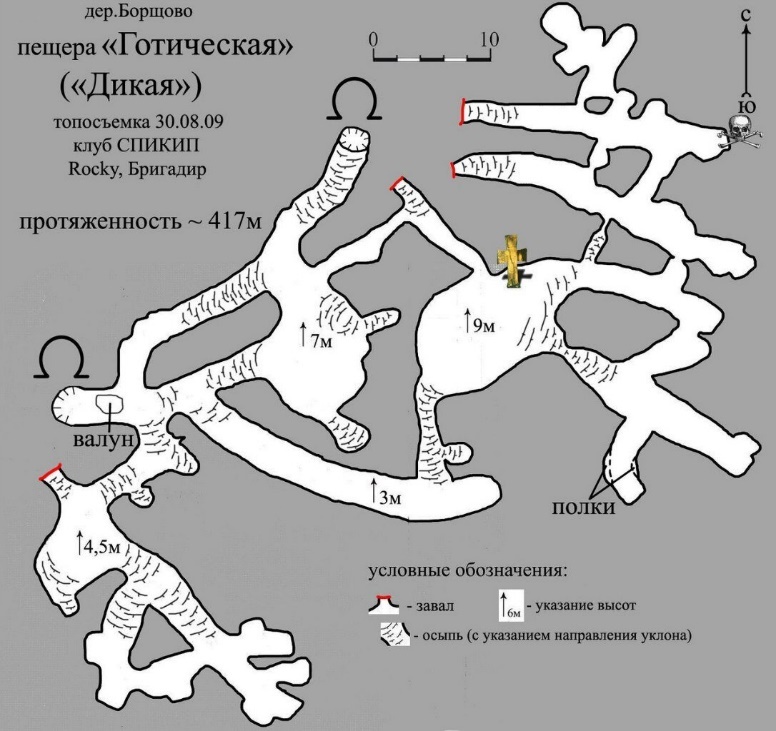
Естественный лабиринт – это природный элемент, созданный либо стихией, либо животными, использующими лабиринт в качестве своего дома. Например, грызуны являются подземными жителями, использующими лабиринты. Они роют себе домики с несколькими выходами, чтобы в случае приближения хищника быстро скрыться из вида. В роли тупиков в таких лабиринтах выступают «кладовые» комнаты и «детские». На рисунке 3 представлен план нор крота. Примерно так же выглядят норы хомяков, мышей, сурков и других мелких грызунов.

  
Рисунок 3 – План норы крота

Другим примером естественного лабиринта могут служить муравейники. У них тоже обычно несколько выходов наружу, тупиками являются хранилища ресурсов и комнаты с потомством. План муравейника в разрезе представлен на рисунке 4.

  
Рисунок – План муравейника

Искусственные лабиринты – лабиринты, созданные человекам в личных целях. К ним можно отнести пещеры, созданные для добычи ресурсов. На рисунке 5 представлен план пещеры в деревне Борщово, которая в XIX веке была одним из основных источников кварца в Санкт-Петербургской губернии.

  
Рисунок 5 – План пещеры

Другим примером искусственных лабиринтов можно считать садовые лабиринты, созданные для украшения приусадебных участков, парков и площадей. На рисунке 6 показан лабиринт, построенный в 2016 году, для принца Уэльского в Дамфрис-хаусе.

  
Рисунок 6 – Лабиринт в Дамфрис-хаусе

* + - 1. Класс топологии описывает геометрию пространства, в котором расположен лабиринт.

Бывают обычные лабиринты – лабиринты в эвклидовым пространстве, и необычные. Примерами лабиринтов с необычной топологией могут быть лабиринты на поверхности куба, лабиринты на поверхности ленты Мёбиуса и лабиринты, эквивалентные находящимся на торе, где попарно соединены левая и правая, верхняя и нижняя стороны. Пример лабиринта с необычной топологией представлен на рисунке 7.

  
Рисунок 7 – Пример лабиринта с необычной топологией

* + - 1. Тесселяция – классификация геометрии отдельных ячеек лабиринта. Существует огромное множество различных форм ячеек лабиринта, среди которых есть как стандартные формы: прямоугольные, треугольные, шестиугольные, так и необычные.

Дельта-лабиринты состоят из соединенных треугольников, при этом у каждой ячейки может быть до трех соединенных с ней проходов. Пример такого лабиринта представлен на рисунке 8.

  
Рисунок – Дельта-лабиринт

Гексагональные лабиринты составлены из шестиугольников, и у каждой ячейки может быть не более пяти проходов. Пример такого лабиринта представлен на рисунке 9.

  
Рисунок – Гексагональный лабиринт

Тета-лабиринты состоят из концентрических окружностей проходов, в которых начало или конец находится в центре, а другой — на внешнем крае. Пример такого лабиринта представлен на рисунке 10.

  
Рисунок – Тета-лабиринт

К необычной тесселяции можно отнести дзета-лабиринт, расположенный на прямоугольной сетке, но у которого помимо прямоугольных соединений ячеек, есть диагональные проходы под углом 45 градусов. Пример такого лабиринта представлен на рисунке 11.

  
Рисунок – Дзета-лабиринт

Фрактальный лабиринт – лабиринт, составленный из лабиринтов, в каждой его ячейке расположен лабиринт меньшего размера. Содержимое копирует себя, создавая бесконечно большой лабиринт. Пример такого лабиринта представлен на рисунке 12.

  
Рисунок – Фрактальный лабиринт

* + - 1. Классификация по маршрутизации связана с типами проходов лабиринта в пределах его геометрии.

Лабиринт с одиночным соединением, он же «идеальный» лабиринт – это лабиринт, в котором ровно один путь к любой другой точке. В нем нет петель, замкнутых цепей, недостижимых областей. Существует только одно решение такого лабиринта. Пример «идеального» лабиринта приведен на рисунке 13.

  
Рисунок – «Идеальный» лабиринт

Лабиринт с многократными соединениями или «плетеный» лабиринт – это лабиринт, в котором не тупиков. Он состоит из проходов, замыкающих и возвращающихся друг к другу, создавая петли вместо конечных точек. Пример приведен на рисунке 14.

  
Рисунок – «Плетеный» лабиринт

Одномаршрутный лабиринт – лабиринт без развилок, в котором есть только один длинный извивающийся проход, меняющий направление. Потеряться в таком лабиринте можно только от невнимательности или усталости. Пример одномаршрутного лабиринта приведен на рисунке 15.

  
Рисунок – Одномаршрутный лабиринт

* + 1. Описание алгоритмов генерации лабиринта

При создании лабиринтов требуется применять общие правила: не должно быть замкнутых контуров или петель, изолированных от других частей лабиринта областей, количество входов и выходов для неидеальных лабиринтов должно быть определено. Для генерации лабиринтов существуют множество алгоритмов [7]. Рассмотрим самые популярные из них.

* + - 1. Алгоритм Эллера. Данный алгоритм основан на построчной генерации, где между каждыми двумя клетками строки при определенных условиях случайным образом возникала стенка. Данный метод гарантирует отсутствие циклов и недоступных клеток посредством выполнения условий. В конце все клетки окажутся «в одном множестве», что будет означать, что между каждыми двумя клетками существует путь.
      2. Алгоритм на основе двоичных деревьев. Для каждой ячейки существует проход вверх или влево, но никогда не в обоих направлениях. В версии с добавлением стен для каждой вершины добавляется сегмент стены, ведущий вниз или вправо, но не в обоих направлениях. Каждая ячейка независима от всех других ячеек, за неимением нужды при её создании проверять состояние каких-то других ячеек.
      3. Алгоритм рекурсивного деления. Сначала создается случайная горизонтальная или вертикальная стена, пересекающая доступную область в случайной строке или столбце. Потом вдоль нее случайно размещаются пустые места. Этот процесс рекурсивно повторяется для двух подобластей, сгенерированных разделяющей стеной. Для наилучших результатов нужно добавить отклонение в выборе горизонтали или вертикали на основе пропорций области. Например, область, ширина которой вдвое больше высоты, должна более часто делиться вертикальными стенами.
      4. Алгоритм выращивания леса. Множество ячеек, случайным образом отсортировывается в список «новых». Одна или несколько ячеек перемещаются их из списка «новых» в список «активных». Далее выбирается ячейка из «активного» списка и вырезается проход в соседнюю несозданную ячейку из «нового» списка, добавляя новую ячейку в список «активных» и объединяя множества двух ячеек. Лабиринт завершён, когда становится пустым список «активных».
      5. Алгоритм Олдос-Бродера. В сетке выбирается точка и случайно перемещаемся в соседнюю ячейку. При условии, что осуществилось попадание в невырезанную ячейку, в неё вырезается проход из предыдущей ячейки. Таким образом продолжается движение в соседние ячейки, пока во все ячейки не появится хотя бы один проход. Этот алгоритм создаёт лабиринты с низким показателем текучести.
      6. Алгоритм Краскала. каждые ячейки помечаются уникальным идентификатором, а затем производится обход всех рёбер в случайном порядке. Если ячейки с обеих сторон от каждого ребра имеют разные идентификаторы, то удаляем стену и задаём всем ячейкам с одной стороны тот же идентификатор, что и ячейкам с другой. Если ячейки на обеих сторонах стены уже имеют одинаковый идентификатор, то между ними уже существует какой-то путь, поэтому стену можно оставить, чтобы не создавать петель.

В разрабатываемом проекте будут использованы следующие 2 алгоритма:

* + 1. Описание алгоритмов нахождения пути

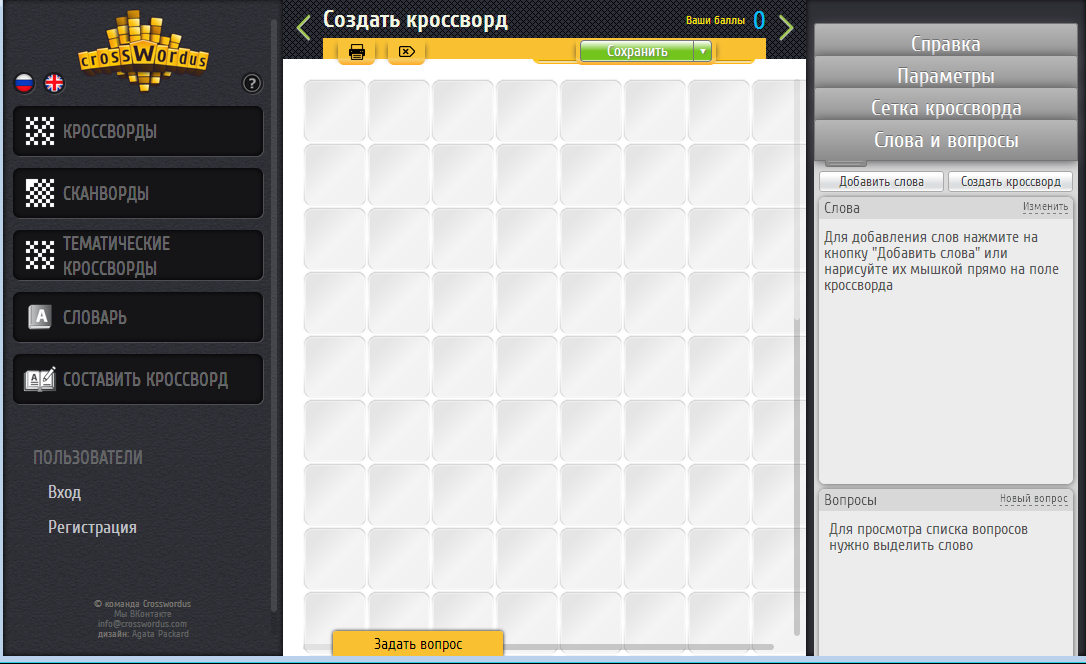
Существует множество способов решения лабиринтов, и каждый из них имеет собственные характеристики. Рассмотрим некоторые из них.

* + - 1. Следование вдоль стен. Начинаем идти по проходам и при достижении развилки всегда поворачиваем направо (или всегда налево). Чтобы применить такое решение лабиринта в реальном мире, нужно положить руку на правую (или левую) стену и постоянно держать её на стене в процессе прохождения лабиринта. При желании можно помечать уже посещённые ячейки и ячейки, посещённые дважды. В конце можно вернуться назад по решению, следуя только по ячейкам, посещённым один раз. Этот метод необязательно найдёт кратчайшее решение, и он совершенно не работает, если цель находится в центре лабиринта и его окружает замкнутая цепь, потому что вы будете ходить вокруг центра и со временем придёте к началу;
      2. Алгоритм цепей. Должны указать нужные места начала и конца, и алгоритм всегда найдёт путь от начала до конца, если он существует. При этом решение склонно быть разумно коротким, если даже не кратчайшим. Это означает, что таким способом нельзя решать лабиринты, в которых неизвестно точное расположение конца;
      3. Волновой алгоритм. Идея этого метода весьма проста: в стороны от исходной точки распространяется волна. Начальное значение волны – ноль. То есть ближайшие точки, в которые можно пойти, например, верх, низ, левая и правая, и которые еще не затронуты волной, получают значение волны, а также некоторый модификатор проходимости этой точки. Чем он больше – тем медленнее преодоление данного участка. Значение волны увеличивается на 1. Обрабатываем аналогично клетки, отходя от тех, на которой значение волны – 2. При этом на клетках с худшей проходимостью волна задержится. И так дальше все обрабатывается, пока не достигнута конечная точка маршрута. Сам путь в получившемся массиве значений волны вычисляется по наименьшим клеткам.
  1. Описание систем-аналогов

Сравнительный анализ аналогичных систем — важный шаг в процессе разработки нового продукта. Он позволяет не только понять сильные и слабые стороны конкурентов, но и определить направления для улучшений и внедрения новых идей. Такой анализ включает несколько ключевых аспектов:

* изучение механик и структуры аналогичных лабиринтов для выявления интересных решений;
* оценка визуальных элементов для анализа стилистических подходов, которые привлекают пользователей;
* сравнение уровней взаимодействия с игроком и степени сложности в разных играх, что помогает определить оптимальные настройки для удержания интереса пользователей.
  + 1. Название системы-аналога 1

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст [4[[3]](#footnote-3)]. На рисунке 16 приведена главная экранная форма программы «Crosswordus», на которой…

  
Рисунок – ПКМ – ИЗМЕНИТЬ ИЗОБРАЖЕНИЕ

К достоинствам данной системы относятся:

* достоинство 1;
* достоинство 1;
* …

К недостаткам системы относятся:

* недостаток 1;
* недостаток 2;
* …
  + 1. Онлайн генератор лабиринтов Plottersvg

В сети Интернет можно найти огромное количество реализаций генерации лабиринтов. которые могут значительно отличаться по функциональности и сложности. Эти реализации варьируются от простых веб-инструментов для создания лабиринтов до сложных программных решений и библиотек, используемых для научных исследований и разработки игр.

Некоторые системы предназначены для демонстрации работы алгоритмов генерации и решения лабиринтов, в то время как другие фокусируются на предоставлении пользователям гибких инструментов для создания уникальных лабиринтов для различных целей.

Рассмотрим один из таких инструментов на сайте https://plottersvg.ru/maze-generator. Интерфейс программы и ее функционал представлен на рисунке 17.



Рисунок – Генератор лабиринтов Plottersvg

К достоинствам данной системы относятся:

* регулирование толщины стенок и размера лабиринта по вертикали и горизонтали;
* три режима расположения входов в лабиринт: по диагонали, слева и справа, сверху и снизу;
* возможность изменения цвета фона, пути и стенок лабиринта;
* возможность сохранения и печати лабиринта.

К недостаткам системы относятся:

* на сайте не представлена информация об алгоритмах, используемых при генерации лабиринта и нахождении пути;
* нет возможности ручной расстановки входов и выходов;
* не предусмотрено ручное прохождение и составление лабиринта.

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

Таблица 1 – Сравнительные характеристики систем-аналогов

|  |  |  |  |
| --- | --- | --- | --- |
| Название системы  Название показателя | Система 1 | Онлайн генератор лабиринтов Plottersvg | Разрабатываемая система |
| Возможность автоматической генерации | + | + | + |
| Автоматическое прохождение |  | + | + |
| Ручное прохождение |  | - | + |
| Оформление по тематике |  | - | + |
| Выбор размера |  | + | + |
| Визуализация прохождения |  | - | + |

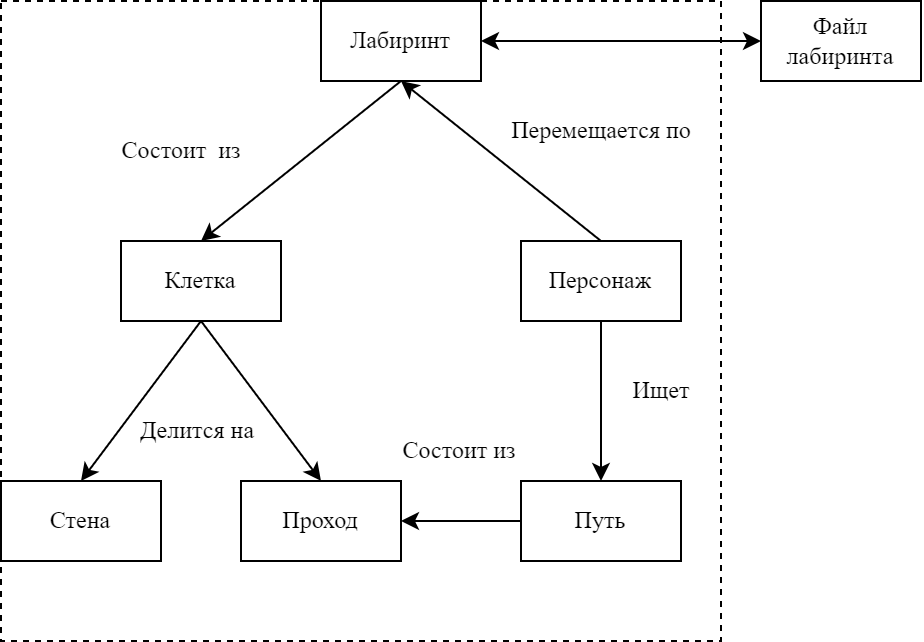
* 1. Диаграмма объектов предметной области

Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова. Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова.

Описание диаграммы

Лабиринт представляет собой совокупность следующих объектов: клетки, которые могут быть стеной или проходом, персонаж, который движется по проходам.

На рисунке 18 приведена диаграмма объектов предметной области. Далее описать основные характеристики объектов.

  
Рисунок 18 – Диаграмма объектов предметной области

* 1. Постановка задачи

Во время курсового проектирования необходимо разработать автоматизированную систему генерирования лабиринта автоматическим способом, а также нахождение выхода из него в автоматическом режиме в соответствии с заданными параметрами. Система должна быть реализована в виде настольного приложения. Система должна быть реализована в виде веб-приложения.

В системе будет предусмотрено две роли пользователей: администратор и игрок. Для работы с системой пользователям необходимо пройти процедуру авторизации: ввести логин и пароль. Система должна аутентифицировать учетные записи, в случае несоответствия выдать предупреждение пользователю.

Постановка задачи пишется на основании приложения к ТЗ (1 часть, функции 5.1 и 2 часть), в повествовательной форме, в будущем времени. Здесь должны быть отражены все основные процессы, которые будут автоматизированы в системе с указанием ограничений, указанных в ч. 1.

В конце должен быть подведен итог.

Таким образом, система должна решать следующие задачи:

1. функции системы:
   * аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
   * генерирование шаблона лабиринта по заданному размеру;
   * случайная расстановка входа и выхода;
   * контроль расстановки входа и выхода;
   * создание структуры лабиринта по заданному алгоритму;
   * проверка структуры лабиринта при загрузке;
   * визуализация генерации лабиринта;
   * визуализация прохождения лабиринта (видимая дорожка);
2. функции администратора:
   * авторизация пользователя в системе (ввод логина и пароля);
   * настройка параметров лабиринта при создании:
3. задание размера по горизонтали;
4. задание размера по вертикали;
5. выбор способа расстановки входа и выхода;
6. выбор алгоритма генерации лабиринта;
7. настройки темы оформления;
   * ручная расстановка входа и выхода;
   * сохранение лабиринта в файл заданной структуры;
   * просмотр лабиринта;
8. функции игрока:
   * регистрация пользователя в системе (ввод логина и пароля);
   * авторизация пользователя в системе (ввод логина и пароля);
   * выбор лабиринта;
   * загрузка лабиринта из файла;
   * изменение темы оформления лабиринта;
   * выбор алгоритма прохождения лабиринта;
   * ручное прохождение лабиринта[[4]](#footnote-4);
   * выбор алгоритма прохождения лабиринта;
   * выбор режима прохождения лабиринта;
   * выбор задержки перемещения персонажа (для автоматического режима);
   * запуск алгоритма.
9. Проектирование системы

Написать, какие задачи решаются на данном этапе жизненного цикла.

* 1. Выбор и обоснование архитектуры системы

Что должно быть отражено

Определение архитектуры

Виды архитектур с кратким описанием

Определение архитектуры клиент-сервер

Виды клиент-серверных архитектур (двухзвенная, трехзвенная с рисунками)

Определение веб-приложения

Определение протокола

Типы протоколов, более подробно про тот, который будет использоваться

Типы клиентов (тонкий/толстый)

Выводы: какая архитектура у вашей системы, какой протокол, какой клиент

* 1. Структурная схема системы

Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова. Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова.

Вводные слова про структурный подход к проектированию программных систем и комплексов программ + определение системы.

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

1. подсистема ???, которая отвечает за …;
2. подсистема ???, которая отвечает за …;
3. …

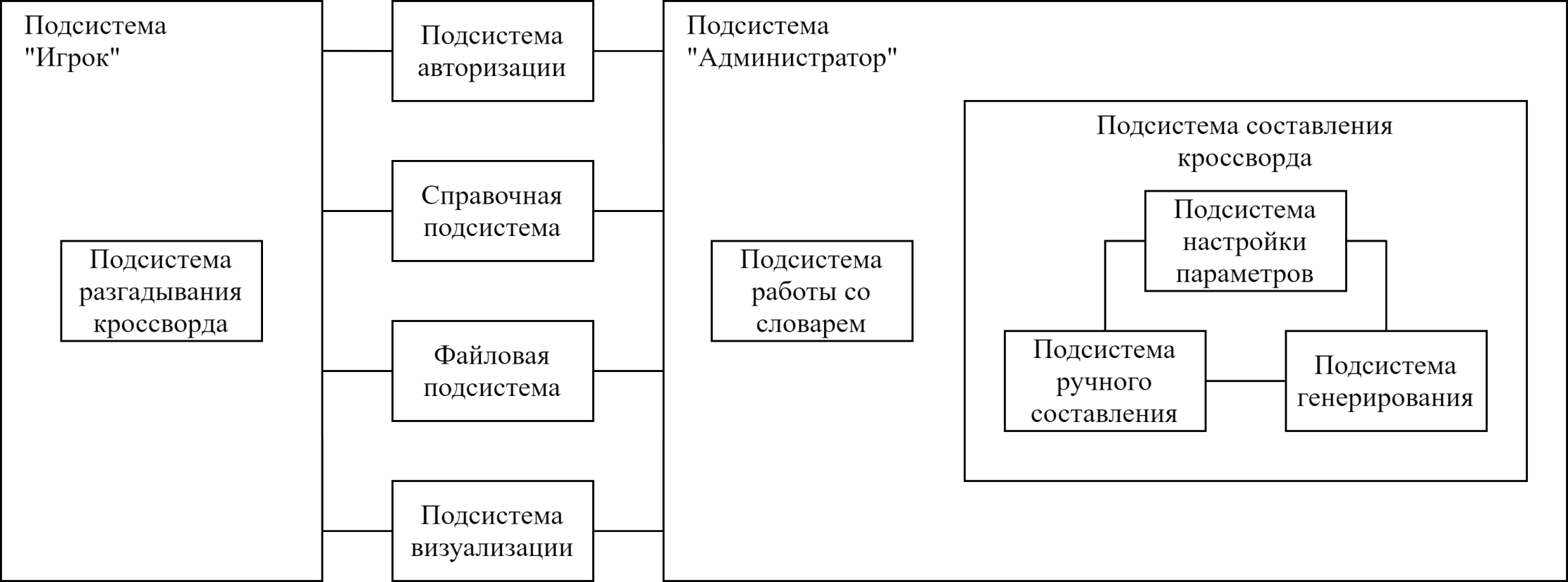
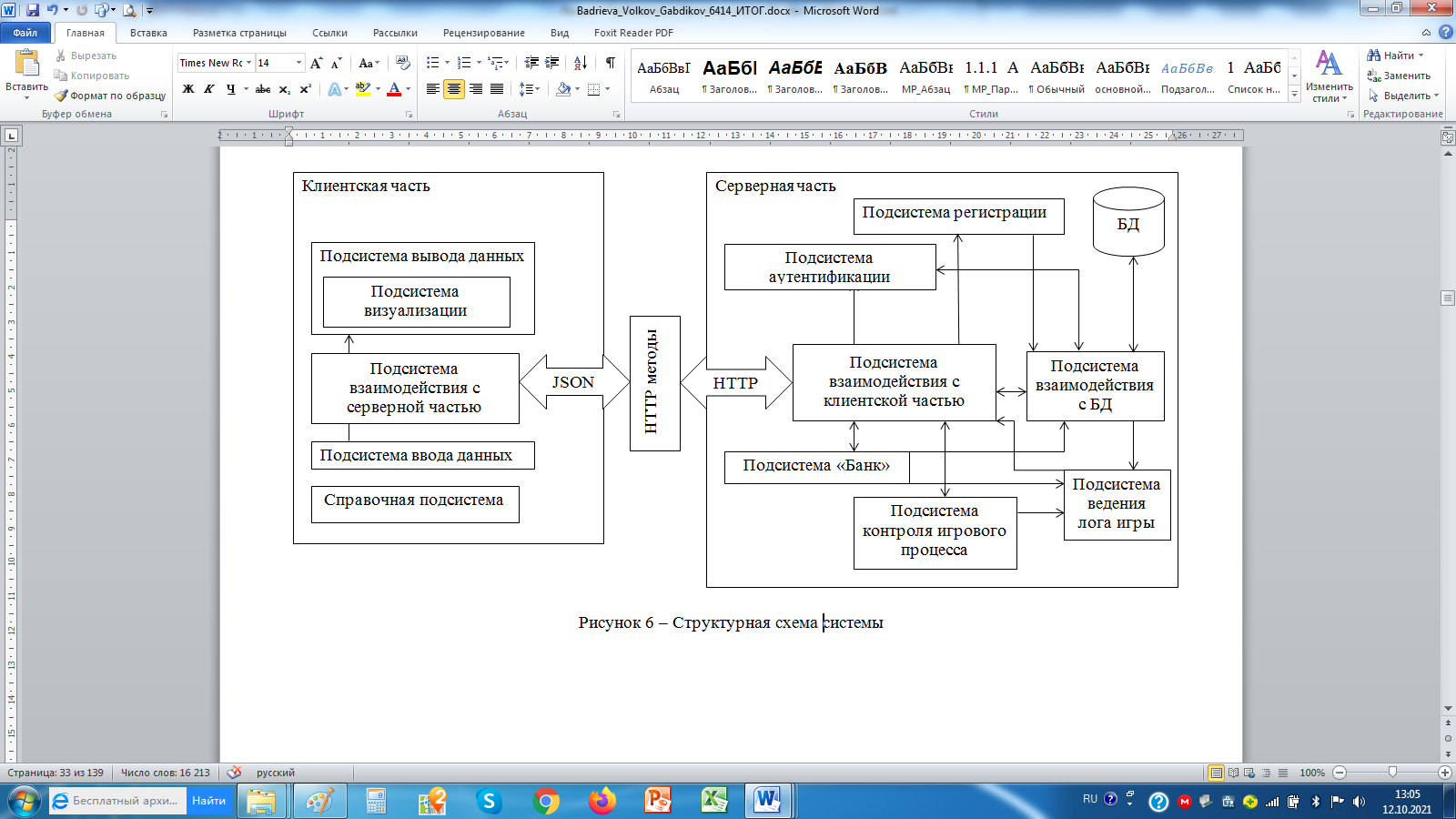
  
Рисунок ххх – Структурная схема системы (для АС составления и разгадывания ЛК)



Рисунок ххх – Структурная схема системы (для АС «Морской бой», технология «толстый клиент»)

  
Рисунок ххх – Структурная схема системы (для АС «Менеджмент», технология «тонкий» клиент»)

* 2. Разработка спецификации требований

Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова. Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова.

* + 1. Функциональная спецификация

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Функциональная спецификация системы приведена в таблице 2.

* + 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.

В таблице 4 приведен перечень исключительных ситуаций для разрабатываемой системы и описаны реакции системы на их возникновение.

Таблица 3 – Перечень функций, выполняемых системой

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название  подсистемы | Название функции | Информационная среда | | | |
| Входные данные | | Выходные данные | |
| Назначение (наименование) | Тип, ограничения | Назначение (наименование) | Тип, ограничения |
|  |  |  |  |  |  |
| Справочная | Выдать сведения о разработчиках | Сведения о разработчиках системы (ФИО, номер группы) | Текст (МЕМО) | Визуальное отображение информации | – |
| Выдать сведения о системе | Файл справки | Текстовый (\*.HTML) |
| Код ошибки | целое |
| Настройки параметров кроссворда | Подключить словарь понятий | Имя файла | Строка, \*.dict | Список понятий и их определений | Динамический массив строк |
| Код ошибки | Целое |
| Задать количество букв в пересечении | Диапазон количества букв | Целое  1..3 | Количество букв в пересечении | Целое |
| Ввести длину кроссворда | Допустимый диапазон значений | Целое  50..300 | Текущая длина кроссворда | Целое |
| Код ошибки | Целое |

Продолжение таблицы 3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| Файловая | Загрузить файл с кроссвордом | Имя файла | Строка, \*.kros | Кроссворд | Объект «Кроссворд», структура определяется в ходе проектирования |
| Код ошибки | Целое |
| Генерирования | Получить список слов | Маска | Строка | Список слов | Динамический массив строк |
| Список понятий и их определений | Динамический массив строк |
| Авторизации | Ввести логин | Набор допустимых символов | Латинские, русские буквы, цифры, «\_» | Логин | Строка |
| Допустимая длина | Целое  4..10 | Код ошибки | Целое |
| Аутентифировать пользователя | Логин | Строка | Код ошибки | Целое |
| Список данных пользователей | Сущность БД «Пользователи» |

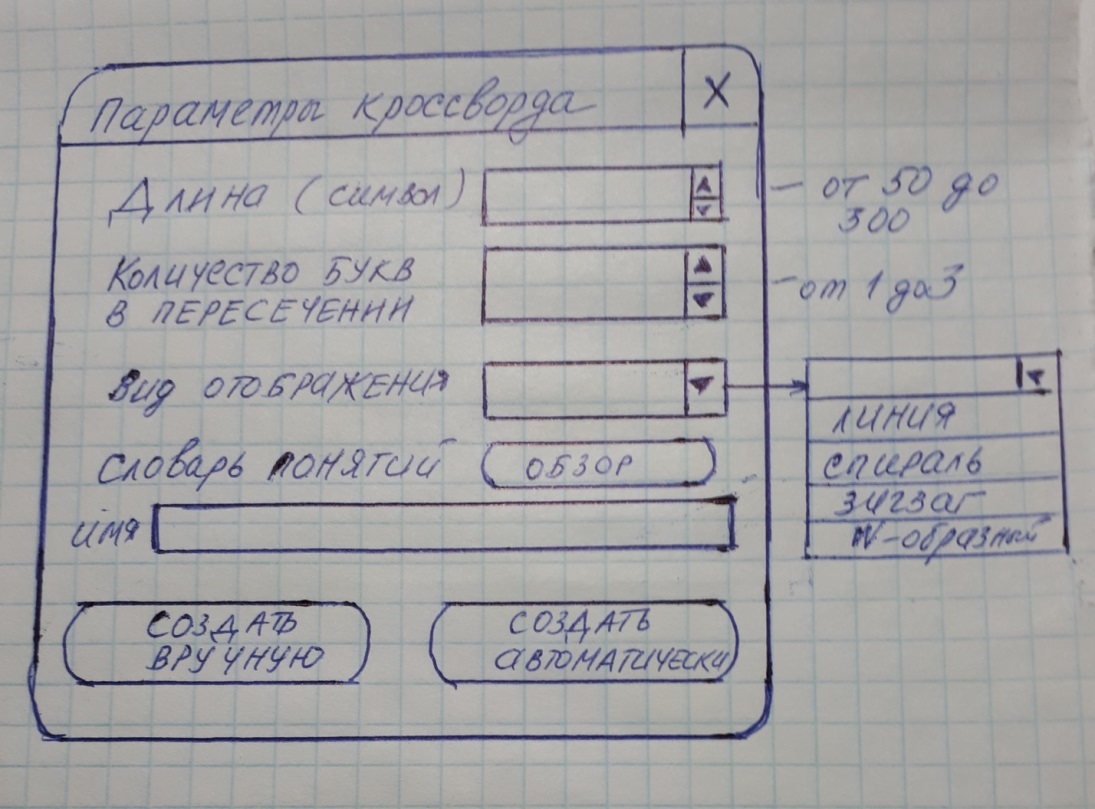
Таблица 4 – Перечень исключительных ситуаций

|  |  |  |
| --- | --- | --- |
| Название  подсистемы | Название исключительной ситуации | Реакция системы |
| 1 Справочная | 1.1 Не возможно открыть файл справки | Выдача сообщения «Файл справки поврежден» |
| 1.2 Не возможно найти файл справки | Выдача сообщения «Отсутствует файл справки» |
| 2 Файловая | 2.1 Попытка открытия файла с несобственным форматом | Выдача сообщения «Файл поврежден или недопустимого формата» |
| … | … | … |

* 1. Разработка прототипа интерфейса пользователя системы

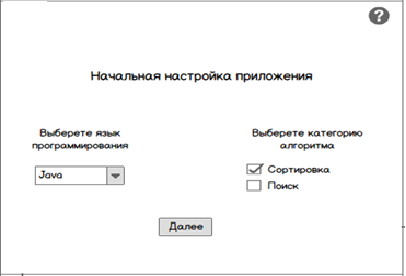
Дать определение интерфейса, отметить основные особенности разработки интерфейса.

Здесь должны быть разработаны прототипы **всех** основных форм приложения с описанием привязанной к ней функциональности, например:

******

***Пример.***

На рисунке ххх приведен прототип экранной формы начальной настройки приложения. Здесь пользователь должен выбрать язык программирования, на котором написан алгоритм, категорию (поиск или сортировка) и нажать кнопку «Далее» для перехода к следующему экрану (форме).

  
Рисунок ххх – Прототип экранной формы начальной настройки приложения

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На Рисунок ХХХ приведена навигационная модель разрабатываемого приложения.

* 1. Разработка информационно-логического проекта системы

Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова.

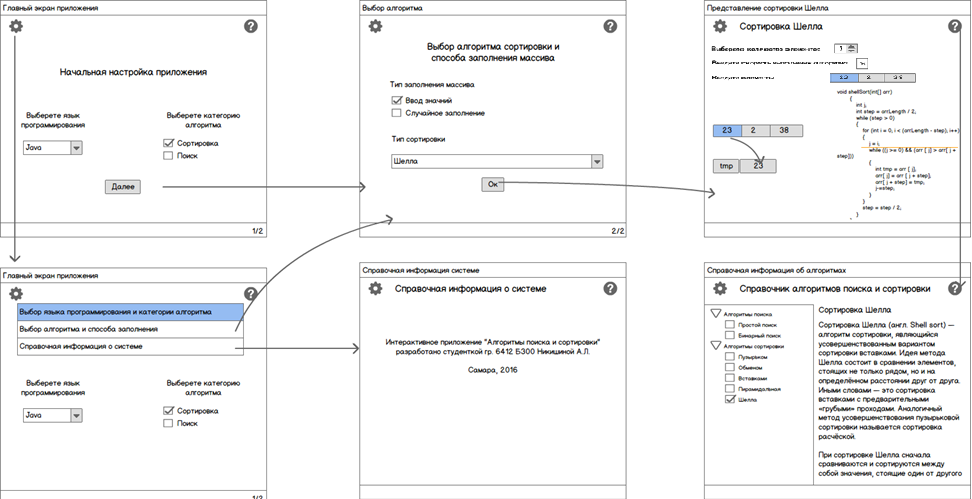
****

Рисунок ХХХ ‒ Навигационная модель приложения

* + 2. Язык UML

Для специфицирования (построения точных, недвусмысленных и полных моделей) системы и ее документирования используется унифицированный язык моделирования UML.

Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные слова.

* + 1. Диаграмма вариантов использования

Диаграмма вариантов использования представляет собой наиболее общую концептуальную модель сложной системы, которая является исходной для построения всех остальных диаграмм. На ней изображаются отношения между актерами и вариантами использования.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма вариантов использования (пользователя). Здесь должно быть описание диаграммы.

* + 1. Сценарии

Сценарий (scenario) ‑ определенная последовательность действий, которая описывает действия актеров и поведение моделируемой системы в форме обычного текста [27].

В контексте языка UML сценарий используется для дополнительной иллюстрации взаимодействия актеров и вариантов использования.

Рассмотрим несколько сценариев.

Сценарии определяются преподавателем.

Рисунок ХХХ − Диаграмма вариантов использования системы

* + 1. Диаграмма классов

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. На практике диаграммы классов применяют для моделирования статического представления системы, они служат основой для целой группы взаимосвязанных диаграмм – диаграмм компонентов и диаграмм размещения [ХХХ].

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

Таблица ХХ – Описание классов системы

|  |  |
| --- | --- |
| Название класса | Назначение |
| 1 | 2 |
|  |  |
|  |  |
|  |  |
|  |  |

* + 1. Диаграмма состояний

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма состояний системы. Здесь должно быть описание диаграммы (диаграмм).

* + 1. Диаграмма деятельности

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма деятельности системы. Здесь должно быть описание диаграммы (диаграмм).

* + 1. Диаграмма последовательности

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграммы последовательности системы для варианта использования «???». Диаграммы построены на основании сценариев, приведенных в п.2.4.3.

* 1. Логическая модель данных (при необходимости)

Логическая информационная модель – модель данных, в которой учитывается способ логического хранения данных в памяти ЭВМ. При построении модели базы данных (БД) используются следующие понятия.

Сущность – объект предметной области, который можно отличить от других понятий по некоторым признакам. Сущность состоит из множества своих экземпляров. Каждая сущность обладает свойствами – атрибутами [**Ошибка! Источник ссылки не найден.**].

Атрибут – определенное свойство сущности. Именно набор атрибутов, в общем случае уникальный для каждой сущности, позволяет выделить ее среди других объектов и назвать уникальным именем.

Атрибут или набор атрибутов, используемый для идентификации экземпляра сущности, называется ключом сущности. В случае если для идентификации экземпляра используется один атрибут, ключ называется простым; в противном случае ключ составной. Каждый экземпляр сущности однозначно определяется ключом [**Ошибка! Источник ссылки не найден.Ошибка! Источник ссылки не найден.**].

Логическая модель БД разрабатываемой системы приведена на рисунке ХХХ.

  
Рисунок 19 – Логическая модель данных

Описание объектов рассматриваемой предметной области, которые хранятся в базе данных, приведено в таблицах 2-???.

Таблица 2 – Сущность «Пользователь»

|  |  |  |
| --- | --- | --- |
| Идентификатор | Тип данных | Описание |
| Ид пользователя | Целый | Уникальный идентификатор пользователя |
| Имя | Символьный[30] | Имя, используемое при идентификации пользователя и его взаимодействии с системой |
| Пароль | Символьный[10] | Пароль пользователя, преобразованный в закодированную строку |
| Email | Символьный[50] | Электронная почта, указанная пользователем при регистрации |

* 1. Выбор и обоснование алгоритмов обработки данных /Разработка и описание алгоритмов обработки данных

Вводные слова про необходимость разработки алгоритмов.

На рисунке ХХХ приведена схема алгоритма обработки элементов массива. Здесь должно быть краткое описание алгоритма.

  
Рисунок ХХХ – Схема алгоритма обработки элементов массива

На рисунке ХХХХ приведена схема алгоритма вычисления исходного выражения.

1

Начало

a

d

result = 0

push eax

eax = a

imul eax

add eax,1

eax = d

ebx = 4

Рисунок ХХХ – Схема алгоритма вычисления исходного выражения (начало)

Рисунок ХХХХ– Схема алгоритма вычисления исходного выражения (окончание)

sub eax, ebx

ebx = eax

eax = 24

pop ebx

idiv ebx

result

Конец

result = eax

idiv ebx

1

* 1. Выбор и обоснование комплекса программных средств

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

* + 1. Выбор языка программирования

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст. Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

* + 1. Выбор среды программирования

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

* + 1. Выбор операционной системы

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

* + 1. Выбор системы управления базами данных (при необходимости)

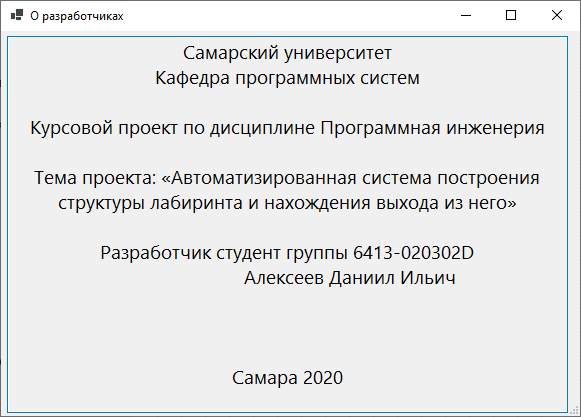
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

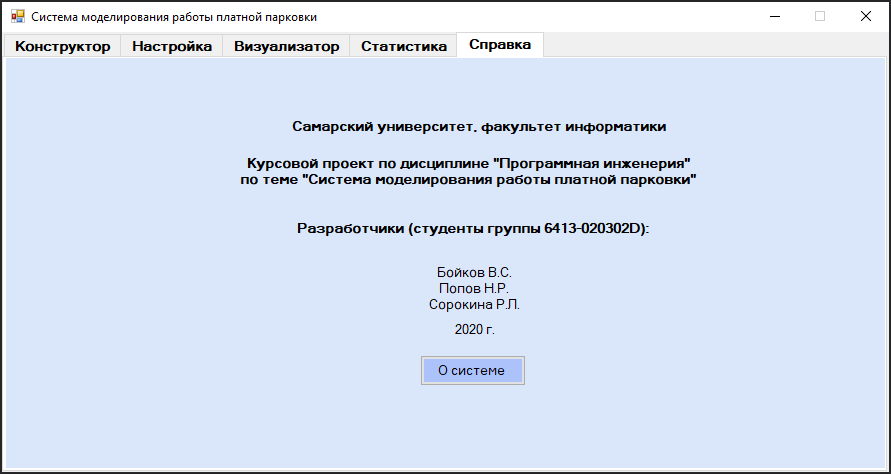
1. Реализация системы
   1. Разработка и описание интерфейса пользователя

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунках приведены примеры того, как нужно оформить сведения о разработчиках.

  
Рисунок ХХХ – Сведения о разработчиках

  
Рисунок ХХХ – Сведения о разработчиках

* 1. Диаграммы реализации

Диаграммы реализации предназначены для отображения состава компилируемых и выполняемых модулей системы, а также связей между ними. Диаграммы реализации разделяются на два конкретных вида: диаграммы компонентов (component diagrams) и диаграммы развертывания (deployment diagrams) [ХХХ].

* + 1. Диаграмма компонентов

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма компонентов, их описание приведено в таблице ХХХ.

Рисунок ХХХ – Диаграмма компонентов системы

Таблица ХХХ – Описание компонентов системы

|  |  |  |
| --- | --- | --- |
| Название компонента | Назначение компонента | Подсистема |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* + 1. Диаграмма развертывания

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке ХХХ приведена диаграмма развертывания системы. Здесь должно быть описание тех компонентов, которые развернуты на узлах ЭВМ.

Рисунок ХХХ – Диаграмма развертывания системы

* + 1. Диаграмма классов

В соответствии со спецификацией, приведенной в п. 2.5.6, и с учетом выбранного языка программирования (см. п. 2.8.1) разработана диаграмма классов системы (этап реализации), приведенная на рисунке ХХХ.

Рисунок ХХХ – Описание классов системы (этап реализации)

* 1. Физическая модель данных (при необходимости)

Физическое проектирование является последним этапом проектирования базы данных, при выполнении которого принимается решение о способах реализации разрабатываемой базы данных. Во время логического проектирования была определена логическая структура базы данных (которая описывает отношения и ограничения в рассматриваемой прикладной области).

Физическая модель базы данных содержит все детали, необходимые конкретной СУБД для создания базы: наименования таблиц и столбцов, типы данных, определения первичных и внешних ключей [??].

На рисунке **Ошибка! Источник ссылки не найден.** представлена физическая модель данных системы.

Рисунок ХХХ – Физическая модель данных системы

В таблицах ??-?? приведено описание сущностей БД. Первичные ключи выделены жирным шрифтом, а внешние – курсивом.

Таблица ХХХ – Сущность « User »

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Имя атрибута | Тип | Размер (байт) |
| **user Id** | **uniqueidentifier** | **int** | **4** |
| Name | Имя пользователя | varchar(30) | 30 |
| Password | Пароль | varchar(10) | 10 |
| e-mail | Адрес электронной почты | varchar(50) | 50 |
| Размер записи | | | 94 |

* 1. Выбор и обоснование комплекса технических средств
     1. Расчет объема занимаемой памяти

Расчет объема внешней памяти

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

VЖД = VОС + VПР + VСПО + VБД + Vсправки,

где VОС – объем памяти, занимаемый операционной системой (операционная система Windows 7 Professional 64 бит с пакетом обновлений SP1,   
VОС = 20 Гб);

VПР – объем памяти, занимаемый непосредственно файлами приложения (VПР = 2 Мб);

VСПО – объем памяти, занимаемый сопутствующим программным обеспечением (библиотеки cryptopp.dll, simplexlsx.dll, sqlite3.dll, sqlitecpp.dll, Qt Framework 5.11.1, Internet Explorer 9; дадим оценку сверху VСПО в 3 Гб);

VБД – объем памяти, занимаемый базой данных (всеми таблицами) при ее максимальном заполнении. Расчет этой составляющей приведен в таблице ХХХ (VБД = ???? байт = ??? Кб = ??? Мб = ??? Гб).

Vсправки – объем памяти, необходимый для хранения файла справки (Vсправки =0,8 Мб).

Таким образом, суммарный объем внешней памяти составит:

VЖД = 20 Гб + 2 Мб + 3 Гб + ??? Мб + 1 Мб ~ ??? Гб.

Таблица 1 – Расчет объема внешней памяти, необходимой для хранения БД

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица | Размер записи (байт) | Максимум записей | Всего (байт) |
| Пользователь | 94 | 10 | 940 |
| Сотрудник |  | 30 |  |
| Статус сотрудника |  | 10 |  |
| Должность сотрудника |  | 10 |  |
| Место работы |  | 10 |  |
| Кафедра |  | 10 |  |
| ОУ ВО |  | 10 |  |
| Итого | | |  |

Расчет объема ОЗУ

Для расчета необходимого объема ОЗУ воспользуемся следующей формулой:

VОЗУ = VОС + VПР + VБД + Vбраузера,

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

VПР – ОЗУ, которое займет само приложение (не превысит 80 Мб);

VБД – объем данных из базы, который может быть одновременно загружен в оперативную память (дадим ему оценку сверху в 10 Мб).

Vбраузера – ОЗУ, занимаемое браузером (оценим его сверху значением в 100 Мб).

Суммарные объемы ОЗУ составит:

VОЗУ = 2 Гб + 80 Мб + 10 МБ + 100 Мб ~ 2.2 Гб.

Таким образом, 2.2 Гб оперативной памяти можно счесть минимально необходимым для функционирования системы.

* + 1. Минимальные требования, предъявляемые к системе

Для корректного функционирования системы необходимо:

* тип ЭВМ: x86-64 совместимый;
* объем ОЗУ – не менее 3 Гб;
* объем свободного дискового пространства – не менее ??? Гб;
* клавиатура или иное устройство ввода;
* мышь или иное манипулирующее устройство;
* процессор – Intel Pentium не менее 1,5 ГГц;
* дисплей с разрешением не менее 1024 × 768 пикселей;
* операционная система Windows 7 и выше;
* браузер Internet Explorer 9 и выше;
* Qt framework 5.11 и выше.

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта была разработана автоматизированная система …, позволяющая ….

В первом разделе приведены основные понятия предметной области, описаны характеристики систем-аналогов, приведен их сравнительный анализ. На основе проведенного анализа выполнена объектная декомпозиция, отраженная в диаграмме объектов, и сформулирована постановка задачи.

Во втором разделе …

В третьем разделе …

Разработанная система может использоваться …

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Лабиринт [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Лабиринт (дата обращения: 17.09.2024).

Игровые лабиринты [Электронный ресурс]. URL: https://dskrnd.ru/blog/sovety-pokupatelyam/igrovye-labirinty-kakuyu-polzu-dlya-detskogo-razvitiya-oni-prinosyat/#:~:text=%Лабиринт%20–%20это%20не%20только%20интересное,игра%20способствует%20выплеску%20лишней%20энергии (дата обращения: 17.09.2024).

RAD (программирование) [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/RAD\_(программирование) (дата обращения: 17.09.2024).

Краткая история UML [Электронный ресурс]. URL: http://techn.sstu.ru/kafedri/подразделения/1/MetMat/murashev/oop/lec/lec12.htm#:~:text=UML (дата обращения: 17.09.2024).

Буч Г., Рамбо Д., Якобсон А. Язык UML. Руководство пользователя. Изд. 2-е. М.: ДМК Пресс, 2006. 546 с

Лабиринты: классификация, генерирование, поиск решений [Электронный ресурс]. URL: https://habr.com/ru/articles/445378/ (дата обращения: 17.09.2024).

Алгоритмы генерации лабиринтов [Электронный ресурс]. URL: https://tproger.ru/articles/maze-generators?ysclid=m183grujpj942114810+,+https://habr.com/ru/articles/445378/ (дата обращения: 18.09.2024).

**Книги**

***Целиком***

.

*Если нужно указать номера конкретных страниц*

Буч Г., Рамбо Д., Якобсон А. Язык UML. Руководство пользователя. Изд. 2-е. М.: ДМК Пресс, 2006. С. 21.

*Если повторная ссылка на тот же документ*

Буч Г., Рамбо Д., Якобсон А. Язык UML … С. 31.

*Если больше 3 авторов*

Нестационарная аэродинамика баллистического полета/ Липницкий Ю.М. и [др.]. М.: Физматлит, 2003. 176 с.

**Журналы**

Зеленко Л.С., Шумская Е.А. Комплекс программ для работы с учебным контентом в дистанционных обучающих системах// Известия СНЦ РАН. 2015. №2 (5). Т. 17. С. 992-1003.

**Руководящие материалы и ГОСТы**

РД 34.20.571. Методические указания по расчету показателей готовности к работе электростанции и энергосистем. Введ. 1976-10-22. М., 1976. 25 с.

ГОСТ Р 7.0.4-2006. Издания. Выходные сведения. Общие требования и правила оформления. М., 2006. II. 43 с. (Система стандартов по информ., библ. и изд. делу).

**Методические указания или учебные пособия**

Зеленко Л.С. Методические указания к лабораторному практикуму по дисциплине «Программная инженерия». Самара: СГАУ, 2012. 67 с.

**Электронные ресурсы**

Российская гидроэнергетика [Электронный ресурс] // Русгидро: [сайт]. URL: http://www.rushydro.ru/industry/russianhydropower/ (дата обращения: 20.12.2024).

Гидроэлектростанция (гидроэлектрическая станция, ГЭС) // Энциклопедический словарь юного техника М.: Издательство «Педагогика», 1987 [Электронный ресурс] // Библиотекарь.Ру: электрон. библ. 2006-2024. URL: http://www.bibliotekar.ru/enc-Tehnika/58.htm (дата обращения: 20.12.2024).

Субботин А.С. Основы гидротехники [Электронный ресурс]. URL: http://www.cawater-info.net/bk/dam-safety/files/subbotin.pdf (дата обращения: 03.02.2024).

Филиальная структура компании [Электронный ресурс] // Системный оператор Единой энергетической системы: [сайт]. [2009-2017]. URL: http://so-ups.ru/index.php?id=about (дата обращения: 20.12.2024).

Автоматизированные системы управления технологическими процессами гидроэлектростанции [Электронный ресурс] // Микроника. Инжиниринговый центр: [сайт]. [1999-2016]. URL: http://mikronika-energo.ru/products/asutp/ges-asu-tp/ (дата обращения: 24.12.2024).

Автоматизированная система управления производственными процессами [Электронный ресурс] // MEScontrol: [сайт]. [2003-2017]. URL: http://mescontrol.ru/articles/systems (дата обращения: 02.04.2024).

Пушников А.Ю. Введение в системы управления базами данных: учеб. пособие [Электронный ресурс] // CITForum: электрон. библиотека. 1997-2017. URL: https://citforum.ru/database/dblearn/ dblearn06.shtml (дата обращения: 20.12.2024).

Пользовательский интерфейс [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2017. URL: https://ru.wikipedia.org/ wiki/Пользовательский\_интерфейс (дата обращения: 17.03.2024).

***Если необходимо указать системные требования для доступа к документу (наличие специального ПО), то***

Белова С.В. Язык UML. Диаграмма вариантов использования. Систем. требования: PowerPoint. URL: nkse.ru/component/k2/item/  
download/7\_754f5a247edc6ec6be78218f187338a5.html (дата обращения: 17.10.2024).

**Сборники научных трудов или трудов конференций**

Философия культуры и философия науки: проблемы и гипотезы: межвуз. сб. науч. тр./ Саратов. гос. ун-т; [под ред. С.Ф. Мартыновича]. Саратов: изд-во Сарат. ун-та, 1999. 199 с.

Акимова А.Е., Трешников А.А., Зеленко Л.С. Информационная среда ГЭС. Подсистема расчета показателей эффективности работы оборудования // Перспективные информационные технологии (ПИТ-2017): сб. науч. тр. межд. научно-техн. конф.; [под ред. С.А. Прохорова]. Самара: Изд-во СНЦ РАН, 2017. С. 41-44.

***Если электронное издание***

Акимова А.Е., Трешников А.А., Зеленко Л.С. Подсистема расчета показателей эффективности работы оборудования // Математика. Компьютер. Образование: труды XXIV межд. конф., 23-28 января 2017 г., г. Пущино. URL: http://www.mce.su/rus/presentations/ p283063/ (дата обращения: 02.03.2017).

ПРИЛОЖЕНИЕ А  
Руководство пользователя

А.1 Назначение системы

Приводится краткое описание возможностей системы.

А.2 Условия работы системы

***Пример.***

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

1. Требования к техническому обеспечению:

* ЭВМ типа IBM PC;
* процессор типа x86 или x64 тактовой частоты 1400 МГц и выше;
* …

1. Требования к программному обеспечению:

* Windows 7 Professional 64 бит с пакетом обновлений SP1 и выше;
* установленная платформа .Net версии 4.0 и выше;
* установленная СУБД ….

А.3 Установка системы

***Пример.***

Система поставляется в виде zip-архива. Данный файл необходимо распаковать в любую директорию на жестком диске. Запускаемым файлом системы является файл ххх.exe.[[5]](#footnote-5)

А.4 Работа с системой

А.4.1 Работа с системой в режиме администратора (если необходимо)

Вход в систему (авторизация)

…

А.4.2 Работа с системой в режиме пользователя

Вход в систему (авторизация)

Вход в систему (регистрация)

Настройка параметров кроссворда

ПРИЛОЖЕНИЕ Б   
Листинг модулей программы

7-10 страниц исходного кода шрифт Times New Roman 10 пт 1 интервал

1. Необязательно для реализации [↑](#footnote-ref-1)
2. *Количество страниц, рисунков, таблиц указывается с учетом приложений* [↑](#footnote-ref-2)
3. Должен быть указан адрес сайта, на котором размещена программа [↑](#footnote-ref-3)
4. Необязательно для реализации [↑](#footnote-ref-4)
5. Если необходимы дополнительные ресурсы для обеспечения работоспособности системы, то все для них также должны быть перечислены условия установки. *Если установка нестандартная, то она должна быть подробно описана (в объеме, достаточном для понимания пользователя).* [↑](#footnote-ref-5)