МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ
ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ЛУГАНСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ЛГПУ»)**

А. А. Тивоненко, О. В. Хитрых

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ
по выполнению курсовой работы

для студентов 1 курса очной и заочной форм обучения
по направлениям подготовки:
44.03.01 Педагогическое образование. Компьютерные системы и образовательная робототехника,
44.03.05 Педагогическое образование (с двумя профилями подготовки). Технология. Информатика

Луганск
Издательство ЛГПУ
2025

УДК 004.42/.43(076)
ББК 32.973.26-018р3
Т39

Р е ц е н з е н т ы :

Остапущенко Д. Л. – доцент кафедры прикладной математики, заведующий МНИЛ «Лаборатория интеллектуальных систем управления» федерального государственного бюджетного образовательного учреждения высшего образования «Луганский государственный университет имени Владимира Даля», кандидат технических наук;

Калайдо А. В. – доцент кафедры технологий производства и профессионального образования федерального государственного бюджетного образовательного учреждения высшего образования «Луганский государственный педагогический университет», кандидат технических наук, доцент;

Дяченко С. В. – доцент кафедры информационных образовательных технологий и систем федерального государственного бюджетного образовательного учреждения высшего образования «Луганский государственный педагогический университет», кандидат педагогических наук, доцент.

Тивоненко, А. А., Хитрых, О. В.
Т39 Методические рекомендации по выполнению курсовой работы / А. А. Тивоненко, О. В. Хитрых ; ФГБОУ ВО «ЛГПУ». – Луганск : Издательство ЛГПУ, 2025. – 42 с.

Методические рекомендации по выполнению курсовой работы по дисциплинам «Основы программирования» и «Программирование» содержат требования к оформлению пояснительной записки курсовой работы, описание типовой структуры пояснительной записки, требования к содержанию разделов, рекомендации к выполнению разделов и подразделов записки, рекомендации по написанию программы, а также критерии оценивания и как будет проходить защита. В приложениях методических рекомендаций приведены примеры оформления: титульной страницы, содержания, списка литературы и разделов записки. Приведен примерный список тем.

Методические рекомендации предназначены для студентов 1 курса очной и заочной форм обучения по направлениям подготовки: 44.03.01 Педагогическое образование. Компьютерные системы и образовательная робототехника, 44.03.05 Педагогическое образование (с двумя профилями подготовки). Технология. Информатика.

УДК 004.42/.43(076)
ББК 32.973.26-018р3

Рекомендовано Учебно-методическим советом ФГБОУ ВО «ЛГПУ»
в качестве методических рекомендаций для студентов очной и заочной форм обучения по направлениям подготовки: 44.03.01 Педагогическое образование. Компьютерные системы и образовательная робототехника, 44.03.05 Педагогическое образование (с двумя профилями подготовки). Технология. Информатика
(протокол № 11 от 17.06.2025 г.)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ..... 4

1. Цели и основные требования, предъявляемые к курсовой работе 5

1.1. Программная среда и исходные данные к курсовой работе 5

1.2. Объем курсовой работы..... 6

1.3. Защита курсовой работы 6

2. Указания к выполнению курсовой работы (написание программы) 7

3. Требования к написанию пояснительной записки..... 9

3.1. Содержание курсовой работы..... 9

3.2. Оформление курсовой работы 15

СПИСОК ИСПОЛЬЗОВАННОЙ И РЕКОМЕНДОВАННОЙ ЛИТЕРАТУРЫ 18

Приложение 1 Образец титульного листа 19

Приложение 2 Образец содержания курсовой работы на тему..... 20

Приложение 3 Образец вариантов тем курсовых работ 21

Приложение 4 Образец пояснительной записки к курсовой работе на тему: «Программа для генерации кроссвордов из имеющих слов и описаний»..... 23

Приложение 5 Образец списка используемой литературы 41

ВВЕДЕНИЕ

Учебные дисциплины «Основы программирования» и «Программирование» относятся к базовой (обязательной) части учебного плана для студентов 1 курса очной и заочной форм обучения по направлениям подготовки: 44.03.01 Педагогическое образование. Компьютерные системы и образовательная робототехника и 44.03.05 Педагогическое образование (с двумя профилями подготовки). Технология. Информатика.

Целью освоения дисциплин является формирование знаний, умений и навыков, необходимые для написания программ на языке C++, рационального использования среды программирования Visual Studio 2022 и выше при обработке разнообразных задач, связанных с обработкой информации, расчетами, поиском, сортировкой данных с применением современных методов и технологий программирования.

Задачами курса являются:

– ознакомление с основами алгоритмизации;

– ознакомление с принципами структурного программирования;

– ознакомление со способами создания алгоритмов, написания программ на языке программирования C++;

– предоставление теоретических знаний и практических навыков по написанию и отладке программ в Visual Studio 2022 и выше.

В рамках курса обучающиеся должны получить навыки использования базовых понятий и определений структурного программирования, разработки программного обеспечения с применением современных методов и технологий программирования. Заканчивается курс написанием курсовой работы, состоящей из практической части – написания программы и теоретической части – написания пояснительной записки.

Базой для выполнения курсовой работы являются знания и умения, полученные в первом и втором семестре первого курса обучения.

Методические рекомендации содержат инструкции по написанию практической части курсовой работы, описание требований к оформлению и содержанию пояснительной записки к курсовой работе, а также описание примера выполнения пояснительной записки курсовой работы.

Курсовая работа выполняется под руководством преподавателя, читающего дисциплины «Программирование» и «Основы программирования». Оценка работы производится членами комиссии, в которую входят руководитель курсовой работы и преподаватели, ведущие лабораторные занятия, которые совместно оценивают соответствие работы заданию и качество самой работы и ее доклада (презентации).

Таким образом, курсовая работа готовит студента к решению его основной задачи – приобретению знаний, умений и навыков самостоятельной работы программиста.

Примерные темы курсовых работ представлены в Приложении 3.

1. Цели и основные требования, предъявляемые к курсовой работе

Курсовая работа по дисциплинам «Программирование» и «Основы программирования» предполагает знание студентами основных принципов написания программ, составления алгоритмов и использование в своей работе уже известных, также знание среды программирования и знание синтаксиса языка C++, и имеет следующие цели:

– развитие у студентов самостоятельности в написании кода и творческих способностей в создании алгоритмов, углубления и закрепления теоретических знаний в разработке программ;

– приобретение практических навыков использования среды программирования;

– приобретение навыков написания кода на языке C++;

– закрепление умений пользоваться справочной литературой, верно оформлять программную документацию.

Курсовая работа выполняется студентом в соответствии с заданием, которое составляется преподавателем или преподавателем совместно со студентом и выдается студенту.

В задании на курсовую работу указывается ее тема и пояснение, включающее в себя техническое задание и исходные данные, объем работы, литературу, сроки выполнения работы по графику.

Содержание курсовой работы, последовательность и объемы выполняемых этапов изложены в разделах 2 – 3.

1.1. Программная среда и исходные данные к курсовой работе

Основное требование для написания программной части курсовой работы – это использование языка C++. Рекомендуемая среда для разработки программы – Microsoft Visual Studio 2022, тип проекта – консольное приложение. Для отладки программы на C++ в Visual Studio используется встроенный отладчик – Microsoft Visual Studio Debugger. Он входит в состав всех версий программы и предназначен для анализа выполнения кода, проверки переменных и устранения ошибок. В случае невозможности использования рекомендуемой среды, разрешается использовать другую, предварительно обсудив свой выбор с руководителем курсовой работы.

Исходными данными являются:

– литературные источники по данной тематике: учебно-методическая литература и интернет страницы, посвященные рассматриваемой теме;

– справочная информация по синтаксису языка C++ и Microsoft Visual Studio 2022, которая может являться базовой при разработке программы.

1.2. Объем курсовой работы

Курсовая работа состоит из разработки программного приложения на языке C++, ее тестирования, а также описания в пояснительной записке процессов создания и тестирования программы и принципов работы с ней. Выполнение курсовой работы включает следующие этапы:

1. Анализ цели и постановки задачи на курсовую работу. Обзор методов решения подобных задач и их анализ. Выбор метода (или альтернативных методов) решения поставленной задачи;
2. Формализация задачи. Аналитическое представление задачи в виде ключевых моментов (действий) и математических формул (если таковые необходимы);
3. Разработка алгоритмов для реализации поставленной задачи;
4. Написание программы, реализующей написанные ранее алгоритмы;
5. Отладка программы и исправление ошибок, тестирование полученной программы;
6. Написание пояснительной записки.

Пояснительную записку рекомендуется составлять во время написания и тестирования программы, т.е. по разделам, а в конце только отформатировать текст согласно требованиям и стандартам.

1.3. Защита курсовой работы

Студент, выполняющий курсовую работу, обязан посещать консультации согласно установленному расписанию, независимо от проделанной работы. В сроки, указанные в задании, студент обязан представить преподавателю материал о проделанной работе.

За неделю до защиты курсовой работы студент обязан сдать окончательный вариант пояснительной записки в отпечатанном и электронном виде и программу в виде исходного кода и исполняемого файла с набором тестовых случаев для проверки корректной работы, получившегося программного продукта.

Дата защиты курсовой работы определяется директоратом, а состав комиссии, которая принимает защиту – кафедрой, ведущей указанные дисциплины. Обычно комиссия состоит из: ведущего преподавателя, заведующего кафедрой, руководителя работы, причем ведущий преподаватель и руководитель работы может быть одним и тем же человеком.

Защита проводится в аудитории с возможностью демонстрации полученного программного продукта. Студент отчитывается о проделанной работе, демонстрирует свою программу и описывает принципы ее работы. По результатам доклада студента, основываясь на отзывы руководителя и рекомендации комиссии, выставляется оценка за проделанную курсовую работу. На доклад и демонстрацию программы каждому студенту отводится не более десяти минут, а затем задаются вопросы по самой работе. Для доклада разрешается использовать презентацию.

При составлении доклада основное внимание студентам необходимо обратить на следующие пункты:

1. Тема курсовой работы.
2. Постановка задачи (техническое задание).
3. Описание алгоритмов и демонстрация программы, с указанием основных моментов написания кода программы.
4. Выводы по итогам проделанной работы.

А защита включает в себя:

1. Демонстрацию выполнения программы на тестах и/или контрольном примере, подготовленных заранее.
2. Демонстрацию исходного кода.
3. Ознакомление преподавателя с пояснительной запиской.
4. Ответы на вопросы преподавателя (например, «почему было реализовано именно таким образом», «имело ли смысл предусмотреть в программе такие-то функции» и т.п.).

2. Указания к выполнению курсовой работы (написание программы)

Первым этапом выполнения курсовой работы является анализ полученного задания, т.е. необходимо изучить предметную область к которой относится задача, изучить имеющиеся аналоги, определить основные принципы и моменты работы будущей программы, определить какие входные данные должны быть, что должно получиться на выходе, какие типы данных необходимо использовать, каким образом должен быть построен диалог с пользователем. Оцениваются возможные подходы к решению поставленной задачи.

Вторым этапом является обзор имеющихся алгоритмов, которые позволяют решить поставленную задачу или подзадачу курсовой работы. Производится их анализ и выбор одного или нескольких алгоритмов, также строится общий алгоритм работы будущей программы. Если имеющиеся алгоритмы не подходят или не устраивают по каким-либо причинам, то происходит разработка собственного алгоритма для решения поставленной задачи. Также на этом этапе необходимо вручную пройти все шаги будущей программы, т.е. сымитировать процесс ее работы – это позволит лучше понять принципы ее функционирования.

Третьим этапом является написания самого кода программы и ее тестирования (отладки). Если компилятор будет сообщать об ошибках в коде, их необходимо устранять самостоятельно, руководствуясь справочной литературой и советами преподавателя на консультациях. В случае возникновения ошибок в процессе тестирования, а именно уже в готовом программном продукте, придется возвращаться к предыдущим этапам, так как это может быть связано с: какими-либо алгоритмическими недоработками или неправильным пониманием задания.

Все переменные, константы, функции должны быть названы в соответствии со смысловой нагрузкой задания.

Меню (диалог) с пользователем, если предусмотрено заданием и программой, должен быть на русском языке.

Для написания программы используется язык программирования C++, интегрированная среда разработки программного обеспечения Microsoft Visual Studio 2022, тип проекта – консольное приложение (console application).

Требования к программной документации (тексту программы):

– каждый программный блок должен иметь комментарии, которые содержат описание назначения функций и методов, перечень функций, которые их вызывают, состав входных и выходных данных (параметров);

– функциональные комментарии должны быть сжатыми, четкими, исключающими возможность неоднозначного толкования, в них должна соблюдаться мнемоника (смысловая нагрузка) имен переменных;

– для локальных и глобальных переменных необходимо указать их назначение и характеристику;

– в структуре текста программы должно отображаться вложение выполняемых действий, т.е. текст программы должен быть читаемым с точки зрения вложенных циклов и условий.

Для написания программы необходимо знать темы изучаемых дисциплин. Они с коротким содержанием предоставлены ниже:

Синтаксис языка C++. Переменные и константы

Элементы и структура языка. Переменные глобальные и локальные. Константы. Главная функция программы в консольном представлении. Арифметические операции.

Типы данных. Преобразование типов

Стандартные типы данных и работа с ними.

Условный оператор

Оператор if ... else. Синтаксис, назначение. Алгоритмы с ветвлением.

Оператор множественного выбора switch

Оператор switch ... case. Синтаксис, назначение.

Циклы с пред- и постусловием. Цикл for. Операторы break и continue

Оператор while, do… while. Синтаксис, назначение. Циклические алгоритмы. Оператор for. Синтаксис, назначение. Прерывание цикла, пропуск итерации. Циклические алгоритмы.

Одномерные массивы.

Работа с одномерными массивами. Заполнение массивов. Поиск элементов. Замены элементов.

Двумерные массивы.

Работа с двумерными массивами. Заполнение массивов. Поиск элементов. Замены элементов.

Строки. Массивы строк

Библиотека для работы со строками string.

Функции (объявление, вызов, аргументы функции)

Типы функций. Объявление, вызов, аргументы функции. Передача аргументов по адресу, по значению. Константные аргументы.

Сортировки

Сортировки: пузырьковая, выбором, вставками.

Поиск данных в массивах

Алгоритмы поиска: линейный и бинарный.

Структуры. Массивы структур

Структуры – как пользовательский тип данных. Массивы структур.

Указатели. Ссылки

Определение. Назначение. Применение.

Динамическая память. Операторы new и delete

Определение. Назначение. Применение.

Одномерные и двумерные динамические массивы

Работа с динамическими массивами. Библиотеки vector, map.

Работа с файлами в C++

Библиотека для работы с файлами fstream.

3. Требования к написанию пояснительной записки

3.1. Содержание курсовой работы

Курсовая работа содержит пояснительную записку на русском языке и электронную версию компьютерной программы. Общий объем пояснительной записки курсовой работы должен составлять примерно 25 страниц печатного текста, но не менее 22 и не более 30, и она должна по своему содержанию давать достаточно полное представление о принципах и методах решения поставленной задачи с обоснованием правильности выбора алгоритма, описанием полученных функций программы и проведенного тестирования. Все материалы, полученные в процессе выполнения курсовой работы, оформляются в соответствии с требованиями и готовятся к защите. Для написания пояснительной записки используется программа MS Word 2016 и выше.

Пояснительная записка должна содержать в указанной последовательности следующие основные разделы:

– титульный лист;

– реферат (аннотация);

– содержание с указанием страниц;

– введение;

– разделы и подразделы основной части (Теоретическая часть (Постановка задачи и предметная область, Эскизный проект, Алгоритмы программы), Практическая часть (Функции программы, Тестирования и интерфейс программы));

– заключение (вывод);

– список используемой литературы;

– приложения (листинг программы с комментариями, блок-схемы).

Титульный лист должен соответствовать установленному образцу (Приложение 1). Важно, что на нем номер страницы не ставится, его можно напечатать отдельно и начать нумерацию со второй страницы в записке, а можно вместо разрыва страниц поставить разрыв раздела. Для этого необходимо перейти в «Параметры страницы» на вкладке «Макет» и выбрать из списка «Разрывы» из подсписка «Разрывы разделов»: «Следующая страница». На странице, на которой поставили разрыв, номер не нужен, поэтому удаляем его. А на следующей странице переходим в нижний колонтитул, щелкая мышкой дважды на него, и на вкладке «Конструктор» отключаем функцию «Как в предыдущем разделе», и выставляем номера страниц, начиная со второй.

Реферат (Аннотация) кратко раскрывает содержание работы и является обязательным разделом в курсовой работе. В содержание не входит. В нем должны излагаться краткие сведения о проведенной работе. Реферат строится по следующей схеме:

– данные об объеме пояснительной записки, количестве иллюстраций, таблиц, использованных источников (в том числе на иностранных языках и интернет-ресурсов) и приложений;

– перечень ключевых слов (от 5 до 15 слов) или словосочетаний из текста пояснительной записки, которые в наибольшей мере характеризуют ее содержание и обеспечивают возможность информационного поиска. Ключевые слова пишутся в именительном падеже и прописными буквами.

Объем текста реферата не должен превышать половину листа А4.

Содержание включает наименование всех разделов и подразделов курсовой работы с указанием номеров страниц начала этих пунктов, включая литературу и приложения. Содержание начинают с нового листа. Названия разделов записывают строчными буквами, начиная с прописной буквы. Пример содержания курсовой работы представлен в Приложении 2.

Содержание делается автоматически, с помощью вкладки «Ссылки» и пункта «Оглавление» MS Word. Для создания автоматического оглавления заголовки разделов и подразделов пояснительной записки необходимо делать с использованием стилей (заголовок1, заголовок 2 и т.д.) на вкладке «Главная». Содержание включают в общее количество листов пояснительной записки.

Введение также является обязательным разделом пояснительной записки курсовой работы и содержит краткий анализ целей и назначения разработки программы, объясняет причины выбора данной тематики, и объясняет выбор среды разработки и выбор языка программирования для написания программной части задания. Во введении дается краткий анализ возможных методов решения поставленной задачи, а также анализируются ограничения и требования к программе. Введение должно кратко характеризовать современное состояние вопроса, которому посвящены работа и цель разработки, отражать существующие и новые решения вопроса, рассматриваемого в работе. Введение начинается с нового листа и занимает одну – две страницы.

Основная часть пояснительной записки состоит из двух разделов, которые называются: «Теоретическая часть» и «Практическая часть».

Теоретическая часть состоит из трех подразделов: «Постановка задачи и предметная область», «Эскизный проект», «Алгоритмы программы». Сама часть начинается с новой страницы, а подразделы – нет. Для того чтобы вставить разрыв страницы в MS Word необходимо поставить курсор в место разрыва и перейти на вкладку «Вставка» и выбрать пункт «Разрыв страницы» или нажать сочетание клавиш Ctrl+Enter.

В разделе «Постановка задачи и предметная область» дается общее описание поставленной задачи, определяется класс задачи и специфические особенности реализации этого типа задач. Должны быть указаны для чего будет использоваться будущая программам и как это будет достигнуто.

Необходимо сформировать перечень основных требований и правил работы с программой, реализация которых позволит решить поставленную перед студентом задачу. Требования должны быть изложены полно, четко, в терминах, понятных разработчику и пользователю будущего программного продукта. Не разрешается использование формулировок, имеющих неоднозначный смысл. В случае необходимости в текстовую часть могут быть включены примеры и рисунки.

При разработке требований для реализации задания курсовой работы решаются следующие задачи:

– поиск информации, необходимой для выполнения функций программы;

– обеспечение полноты и четкости выполнения функций;

– выявление пространственно-временных ограничений, налагающихся на программу;

– обнаружение пунктов задачи, которые со временем могут быть изменены или улучшены.

Эскизный проект содержит, в основном, только внешние атрибуты, то есть подробное описание того, какой вид будет иметь программа с точки зрения пользователя и диалога с ним, не касаясь вопроса практической реализации. На этой стадии уточняются и даются в более формализованном виде требования к программе, определяются ее функции и проектируется взаимодействие с пользователем, разрабатываются форматы входных данных и выходной информации. В этом разделе необходимо проанализировать сформированные на предварительном этапе требования и составить:

– краткое описание программы;

– описание пользователей приложения;

– подробное описание функциональных задач;

– перечень функций, которые должны быть разработаны для выполнения функциональных задач;

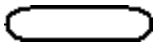
– описание методов решения задач и требований к программной среде для разработки системы.

В разделе «Алгоритмы программы» приводится поэтапное описание каждого действия, которое должно выполняться программой, а именно

описание алгоритмов словесное и/или графическое, используемых в программе. Если программа использует общеизвестные алгоритмы и программные решения, необходимо предоставить информацию о первоисточнике. Блок-схемы алгоритмов могут располагаться прямо в этом разделе или в графических приложениях (если очень большие).

Графические объекты, используемые для составления блок-схем показаны в таблице 3.1.

Таблица 3.1 – Условные обозначения блоков в блок-схемах алгоритмов

Наименование	**Обозначение**	**Действия**
Процесс / действие		Выполнение операции или группы операций, в результате которых изменяется значение, форма представления или расположение данных.
Ввод / вывод		Преобразование данных в форму, пригодную для обработки (ввод) или отображения результатов обработки (вывод).
Условие		Выбор направления выполнения алгоритма в зависимости от некоторых переменных условий.
Предопределенный процесс		Использование ранее созданных и отдельно написанных программ (подпрограмм).
Начало / конец		Начало и конец программы.
Соединитель		Указание связи между прерванными линиями, соединяющими блоки.
Межстраничный соединитель		Указание связи между прерванными линиями, соединяющими блоки, расположенные на разных листах.
Комментарий		Связь между элементом схемы и пояснением.

Создавать блок-схемы можно в MS Word, а можно использовать специальное программное обеспечение Microsoft Visio, yEd Graph Editor, Google Drawings и другие.

Также в этом разделе могут быть описаны: порядок работы с программой, общие необходимые библиотеки и функции.

Практическая часть состоит из двух подразделов: «Функции программы», «Тестирование и интерфейс программы». В разделе «Функции программы» предоставляются части программных функций, написанных студентом, с их полным описанием и описанием входных параметров и возвращаемых

результатов. Стандартные операторы C++ и функции, используемые из библиотек C++, не описываются, а могут быть только перечислены как используемы. Тексты функций обязательно должны иметь комментарии. При использовании в тексте работы готовых описаний алгоритмов и функций необходимо обязательно делать ссылки на соответствующие источники.

Раздел «Тестирование и интерфейс программы» включает в себя описание полученного программного продукта и руководство пользователя этой программой с рисунками, демонстрирующими работу программы. Раздел содержит следующую информацию:

– краткое описание назначения программного продукта и функций, которые он выполняет;

– описание основных характеристик и особенностей программы (режим и время работы, нужный объем памяти);

– ограничения на область применения;

– условия, необходимые для выполнения программной системы;

– краткая инструкция пользователя;

– результаты тестирования.

Заключение является обязательным разделом пояснительной записки и начинается с новой страницы. Оно должно содержать оценку результатов курсовой работы, а именно: насколько выполнено задание, все ли функции реализованы, все ли требования выполнены и все ли результаты были достигнуты. В этом разделе нужно дать четкую характеристику преимуществ и недостатков полученного программного продукта и нюансы его использования. Если присутствует несовпадение полученной программы с полученным ранее заданием, то в заключении необходимо дать объяснения почему так вышло и что планируется предпринять, чтобы получить программу, полностью соответствующую заданию. Если в процессе решения поставленной задачи был избран не лучший, как оказалось, подход, то следует указать причины такого выбора, а также нерешенные вопросы, которые касаются поставленной задачи и рекомендации по возможным путям их решения.

Список используемой литературы также является обязательным разделом пояснительной записки курсовой работы и включается в содержание пояснительной записки. Он предназначен предоставить сведения проверяющему об используемых при написании курсовой работы (программы и пояснительной записки) информационных источниках: книги, статьи, интернет-страницы, видеоролики и т.д. Располагается с новой страницы в виде нумерованного списка. Расположение источников в списке должно следовать в логической последовательности, т.е. по мере использования их в работе и появлений ссылок в тексте. Оформление идет согласно ГОСТ Р 7.0.100-2018. Все источники нумеруют арабскими цифрами. О каждом источнике подаются следующие сведения:

– фамилия и инициалы автора. Если книга написана двумя или тремя авторами, то их фамилии перечисляются по такому порядку, в котором они указаны в книге;

– полное и точное название книги, которая не берется в кавычки;

– подзаголовок, уточняющий название (если он указан на титульном листе);

– данные о повторном издании, название города издание книги в именительном падеже. Для городов Москвы, Санкт-Петербурга (ранее Ленинграда) употребляются сокращения М, СПб (Л);

– название издательства (без кавычек), год издания (без слов «год» или сокращения «г»).

Группы сведений отделяются друг от друга знаком точка и тире (. –). Библиографическое описание делают на языке документа. Документы, имеющие более трех авторов, описывают под названием. При этом за косой чертой (/), которую проставляют после последнего слова названия, приводят инициалы и фамилии авторов. Если на титульном листе отсутствует фамилия автора или авторов, то запись данных о книге начинают с названия книги, после чего за косой чертой указывают фамилию редактора и его инициалы, которые ставят перед фамилией, и все последние элементы по фамилии автора. Сведения о статьях, которые опубликованы в сборниках, журналах и других периодических изданиях, должны иметь фамилию и инициалы автора статьи. Заголовок статьи, после которого идет полное название источника (книги или сборника), в котором находится статья по изложенным выше правилам, а для журнала – название, год выпуска, номера страниц, на которых размещена статья.

При составлении списка использованной литературы необходимо использовать следующую форму:

а) для журнальных статей. Автор(ы), название статьи, одна наклонная (/), еще раз авторы, две наклонных (//), наименование журнала (официальное сокращение), год издания, том, и/или выпуск, и/или номер, тире, страницы.

Например: Кирюхин, В. М., Окулов, С. М. Методика анализа сложных задач по информатике / В. М. Кирюхин, С. М. Окулов // Информатика и образование. 2006. № 4. С. 42–54.

б) для книг с одним или несколькими авторами (до четырех). Автор первый, наименование книги, одна наклонная (/), еще раз все авторы, тире, место издания (сокращенно), двоеточие, издательство (сокращенно), год издания, тире, число страниц.

Например:

1. Павловская, Т. А. C/C++. Программирование на языке высокого уровня / Т. А. Павловская. – Санкт-Петербург : Изд-во Питер, 2009. – 461 с.

2. Дейтел, Х. М. Как программировать на C++ : Пятое издание. Пер. с англ. / Х. М. Дейтел, П. Дж. Дейтел. – М. : ООО «Бином-Пресс», 2008. – 1456 с.

в) для книг с авторским коллективом нескольких авторов (более трех) и под редакцией. Наименование книги, разделитель (/), текст «ред.» и редактор(ы), тире, место издания, двоеточие, издательство, год, тире, число страниц.

Например:

1. Численные методы / ред. М. П. Лапчика. – М.: Academia, 2017. – 608 с.

2. Приемы объектно-ориентированного проектированния. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон. – СПб. : Питер, 2015. – 368 с.

г) для интернет-источников:

Книги в электронных библиотеках: Описание книги соответственно литературных источников, ISBN, тире, Текст: электронный разделитель (//) Цифровой образовательный ресурс, – URL . – Примечание (тип доступа, дата обращения)

Статьи из журналов и сборников: Автор (если есть), заглавие [электронный ресурс] : – Режим доступа: URL. – Примечание (тип доступа, загл. с экрана, дата обращения).

Например:

1. Липпман, С. Язык программирования C++. Полное руководство / С. Липпман, Ж. Лажойе; перевод А. Слинкин. – 3-е изд. – Саратов: Профобразование, 2024. – 1104 с. – ISBN 978-5-4488-0136-5. – Текст: электронный // Цифровой образовательный ресурс IPR SMART: [сайт]. – URL: https://www.iprbookshop.ru/145938.html (дата обращения: 27.12.2024). – Режим доступа: для авторизир. пользователей

2. Программирование роботов: методы и инструменты – [электронный ресурс]. – Режим доступа: https://gb.ru/blog/programmirovanie-robotov/. – Загл. с экрана. – Дата обращения: 31.01.2024.

Образец оформления списка литературы предоставлен в Приложении 5 данных методических рекомендаций.

Делая ссылки на литературу в тексте работы, следует указывать в квадратных скобках порядковый номер книги или статьи по списку использования.

Приложение является также обязательным в пояснительной записке. Оно содержит листинг (код) программы и не входит в общую нумерацию страниц пояснительной записки. По желанию разработчика можно добавить еще необязательные приложения, которые могут содержать вспомогательный материал: дополнительный графический материал (блок-схемы алгоритмов), результаты расчетов, выводы отдельных формул (если они загромождают основной текст), инструкции к использованию программ, таблицы исходных данных. Ссылки на приложения должны быть обязательно в основном тексте записки.

Пример оформления пояснительной записки расположен в Приложении 4.

3.2. Оформление курсовой работы

Курсовая работа должна быть подготовлена на компьютере в среде MS Word и распечатана на бумаге формата А4 (размеры 210×297). Основной текст

набирается на странице с полями: слева – 30 мм, справа – 15 мм, сверху и снизу – 20 мм. Нумерация страниц внизу посередине, причем титульная не нумеруется, хотя и считается за первую. Выставляется с вкладки «Вставка» блок «Колонтитулы».

Шрифт основного текста – Times New Roman 14, для наименований подразделов шрифт Times New Roman 14 полужирный. Для разделов – Times New Roman 16-й шрифт полужирный, заглавными буквами. Интервал – 1,5. Выравнивание по ширине. Расстояние между заголовками раздела и подраздела должно быть равным при компьютерном наборе – 8 мм. Расстояние между заголовком и текстом при компьютерном наборе – 15 мм.

Абзацы начинаются с 6-го символа (отступ 1,25). Нумерация разделов производится арабскими цифрами с точкой (1., 2.). Подразделы нумеруются с дополнительной цифрой (1.2., 2.3.).

Наименования разделов и подразделов начинаются с позиции красной строки и центрируются. В конце наименования раздела и подраздела точка не ставится.

Код программы, вставляемый в текст записки, идет тем же шрифтом Calibri (Основной текст) размер 11.

Списки можно использовать нумерованные (арабскими цифрами), маркированные (неярким и негромоздким символом, например: точка, дефис, тире) и многоуровневые.

Таблицы оформляются отдельным объектом. Наименование таблицы устанавливается по центру таблицы сверху: «Таблица N.M – Название», где N – номер раздела, M – номер таблицы в разделе. В работе все таблицы должны иметь наименование или все таблицы должны быть без наименований. В тексте работы таблицы размещают как можно раньше после их упоминания. Ссылки на таблицы обязательны и имеют вид: «в таблице 1.2.», «см. табл. 1.2.».

При оформлении таблиц большого объема допускается уменьшать размер шрифта до 12 с интервалом 1.

Таблицу стоит помещать после первого упоминания о ней в тексте. Таблицы стоит размещать так, чтобы их можно было читать без поворота страницы. Если такое размещение невозможно, таблицу располагают так, чтобы для чтения надо было записку повернуть по часовой стрелке. Если строки или графы таблицы выходят за формат листа, таблицу делят на части, что, в зависимости от особенностей таблицы, переносят на другие листы или помещают на одном листе рядом, или одну под другой в пределах другой страницы. Если части таблицы помещают рядом, то в каждой части повторяют заголовок. При переносе таблицы на следующий лист пояснительной записки заголовок таблицы следует повторять, и над ней слева помещают слова «Продолжение таблицы» с указанием номера таблицы. В наименовании таблиц точки не ставятся.

Иллюстрации, а именно: скриншоты программы, картинки эскизного проекта, логические схемы программы в пояснительной записке к курсовой

работе должны быть оформлены отдельным объектом (рисунком) и размещаться как можно раньше после ссылки на них.

Блок-схемы алгоритмов должны быть оформлены отдельным объектом (рисунком) и размещаться как можно раньше после ссылки на них, а подписываются также как и рисунки.

Рисунки должны иметь название, подрисуночную подпись, которая центрируется под рисунком. А сама подпись внизу рисунка устанавливается следующим образом: Рисунок N.M – Название, где N – номер раздела, а M – номер рисунка, Название – подрисуночная подпись. В конце подрисуночной подписи точка не ставятся.

Если имеется ссылки на рисунок в тексте, то они имеют вид: «на рисунке 2.1», «см. рис. 2.1».

Страницы в содержании считаются с учетом начала отсчета с титульного листа, но начинают проставлять с реферата.

Формулы готовятся с помощью редактора формул MS WORD по центру строки по всем правилам оформления математических формул. Номер формулы, если он необходим, устанавливается на правой границе строки и имеет структуру (N.M), где N – номер раздела, M – номер формулы в разделе.

Допускается сквозная нумерация таблиц, рисунков и формул для всей работы, и наоборот, более дробная нумерация, с учетом подразделов. В последнем случае структура номера имеет вид (N.M.L.), где N – номер раздела, M – номер подраздела, L – номер таблицы, рисунка или формулы в подразделе.

Приложения оформляют как продолжение основного текста на последующих страницах или в виде отдельной части, располагая их в порядке появления ссылок в тексте. Номера страниц в приложении не ставятся. Достигнуть этого можно, распечатав его отдельно, а можно поставить разрыв раздела. Для этого необходимо перейти на «Параметры страницы» на вкладке «Макет» и выбрать из списка «Разрывы» из подсписка «Разрывы разделов»: «Следующая страница». На новой странице перейти в нижний колонтитул, щелкая мышкой дважды на него, и на вкладке «Конструктор» отключить функцию «Как в предыдущем разделе» и убрать номер страницы. Каждое приложение следует начинать с новой страницы с указанием в правом верхнем углу слова «Приложение 1», «Приложение 2» и т.д. Далее следует тематический заголовок приложения. Шрифт в приложении Calibri, размер 11.

СПИСОК ИСПОЛЬЗОВАННОЙ И РЕКОМЕНДОВАННОЙ ЛИТЕРАТУРЫ

1. Павловская, Т. А. C/C++. Программирование на языке высокого уровня / Т. А. Павловская. – СПб. : Питер, 2021. – 461 с.

2. Прата, Стивен. Язык программирования C++. Лекции и упражнения / Стивен Прата. – М. : ООО «И.Д. Вильямс», 2018. – 1244 с.

3. Доусон, М. Изучаем C++ через программирование игр / Майкл Доусон. – СПб. : Питер, 2016. – 352 с.

4. Скотт, Мейерс Эффективный и современный C++: 42 рекомендации по использованию C++11 и C++14 / Скотт Мейерс. : пер. с англ. – М. : ООО «И.Д. Вильямс». – 2016. – 304 с.

5. Ламберт, Д. Microsoft Word 2016 / Д. Ламберт – М. : Эком, 2019 – 656 с.

6. Зингаро, Даниэль Алгоритмы на практике / Даниэль Зингаро. – СПб. : Питер, 2023. – 432 с.

7. Бхаргава, А. Грокаем алгоритмы. Иллюстрированное пособие для программистов и любопытствующих / А. Бхаргава. – СПб. : Питер, 2017. – 288 с.: ил.

8. Мозговой, М. В. C++ Мастер-класс. 85 нетривиальных проектов, решений и задач / М. В. Мозговой. – СПб. : Наука и техника, 2007. – 272 с.

9. Липпман, С. Язык программирования C++. Полное руководство / С. Липпман, Ж. Лажойе; перевод А. Слинкин. – 3-е изд. – Саратов : Профобразование, 2024. – 1104 c. – ISBN 978-5-4488-0136-5. – Текст: электронный // Цифровой образовательный ресурс IPR SMART: [сайт]. – URL: https://www.iprbookshop.ru/145938.html (дата обращения: 20.12.2024). – Режим доступа: для авторизир. пользователей

10. Методические рекомендации по оформлению рукописей учебных и научных изданий / ФГБОУ ВО «ЛГПУ»; сост. Г. Г. Калинина. – Луганск : Издательство ЛГПУ, 2024. – 88 с.

11. Глузман, Н. А. Выпускная квалификационная работа: подготовка, оформление, защита : учебное пособие / Н. А. Глузман, Н. В. Давкуш, Ю. К. Картавая. – Симферополь : АРИАЛ, 2018. – 198 с.

Приложение 1

Образец титульного листа

МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ
ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ЛУГАНСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ЛГПУ»)

Институт физико-математического образования, информационных и обслуживающих технологий
Кафедра информационных образовательных технологий и систем

КУРСОВАЯ РАБОТА
на тему: _____«*Название*»_____
по дисциплине: _____«*Название*»_____

Студент___ 1-го курса
направления подготовки
*название*_____
профиль *название профиля*
форма обучения *форма*_____
*Ф.И.О. студента*_____

Руководитель: _____*должность*_____
_____*кафедра*_____
_____*Ф.И.О.*_____
Оценка работы _____
Количество баллов _ _ Оценка ECTS _____

Члены комиссии:

_____ _____
(подпись) (фамилия и инициалы)

_____ _____
(подпись) (фамилия и инициалы)

_____ _____
(подпись) (фамилия и инициалы)

Луганск, 20____

Приложение 2

Образец содержания курсовой работы на тему

СОДЕРЖАНИЕ

ВВЕДЕНИЕ …….... 3
1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ …… 6
1.1 Постановка задачи ……. 6
1.2 Эскизный проект …… 9
1.3 Алгоритмы программы ……. 10
2. ПРАКТИЧЕСКАЯ ЧАСТЬ …….. 13
2.1 Разработка функций программы …….. 13
2.2 Интерфейс и тестирование программы …… 19
ЗАКЛЮЧЕНИЕ ……... 23
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ ……. 24
ПРИЛОЖЕНИЕ ……... 25

Приложение 3

Образец вариантов тем курсовых работ

1. Z-функция строки и ее вычисление. Назначение, применение, примеры.
2. Алгоритм сжатия Хаффмана. Назначение, применение, примеры.
3. Алгоритм Кнута – Морриса – Пратта. Назначение, применение, примеры.
4. Алгоритм Манакера. Назначение, применение, примеры.
5. Алгоритм Ахо-Корасик. Назначение, применение, примеры.
6. Алгоритм Бойера-Мура. Назначение, применение, примеры.
7. Алгоритм Ландау-Вишкина (k различий). Назначение, применение, примеры.
8. Игра «Быки и коровы». Компьютер загадывает четырехзначное число с помощью датчика случайных чисел, а игрок отгадывает его, вводя четыре цифры от 0 до 9. Чем меньше попыток было угадать число, тем лучше результат. Если позиция и цифра числа введены верно, тогда это «бык», а если только угадана цифра в числе без указания позиции, тогда это «корова». Пользователь выиграл, если угадал все четыре цифры числа и их позицию. Предусмотреть сохранение статистики результатов игроков.
9. Игра «Судоку». Случайным образом генерировать задачу (саму судоку) и реализовать алгоритм для нахождения ответа. Предусмотреть сохранения результатов генерирования задачи и ответа в файл.
10. Игра «Поле чудес». Из имеющихся загаданных слов компьютер загадывает слово, а пользователь отгадывает, вводя буквы. Предусмотреть сохранение статистики результатов игроков.
11. Игра «Морской бой». Написать игру человека против компьютера. Предусмотреть за человека возможность автоматической (расстановку осуществляет кораблей компьютер случайным образом) и ручной расстановки своих кораблей. Предусмотреть сохранение статистики результатов игроков.
12. Программа «Лабиринт». Поиск пути в сгенерированном по алгоритму лабиринте от входа к выходу с помощью волнового алгоритма. По диагонали ходить нельзя, через препятствия ходить нельзя, движение на один шаг. Размер лабиринта задается пользователем. Препятствия можно генерировать с помощью алгоритма Эйлера или другого алгоритма для решения подобных задач, или случайным образом.
13. Программа «Фильтрация текста». Создать программу, фильтрующую текст из заданного файла. Задача программы заключается в считывании файла и отображении его на экране (постранично), используя замену заданного набора символов на пробелы. Программа должна предлагать следующие варианты наборов символов для фильтрации:

a. Символы латинского алфавита;

b. Символы кириллицы;

c. Символы пунктуации;

d. Цифры.

Фильтры могут накладываться последовательно. При повторной установке существующего фильтра данный фильтр должен сниматься. Написать программу, позволяющую вычисления значения арифметического выражения (ввод в строку), которое может включать в себя действительные числа, а также круглые скобки и следующие операции: +, -, *, /, ^ (возведение в степень). Вычисления должны производиться с учетом скобок и приоритетов используемых операций. Предусмотреть корректную обработку возможных ошибок и информирование о них пользователя.

14. Программа шифровальщик. Разработать программу для шифровки и дешифровки текста с использованием шифра Виженера.

15. Программа шифровальщик. Разработать программу для шифровки и дешифровки текста с использованием решетки Кардано.

16. Программа для генерации кроссвордов из имеющих слов и описаний к ним. Предусмотреть возможность загружать кроссворд из имеющегося файла слов.

17. Игра «Четыре в ряд». Человек играет с компьютером. Предусмотреть возможность первого хода как человеком, так и компьютером.

18. Игра «Тамагочи». Выращивание виртуального питомца с возможностью: кормления, лечения, обучения и т.д.

19. Игра «Ставки». Играющий вводит любое число в диапазоне от 2 до 12 и ставку, которую он делает в этот ход. Программа с помощью датчика случайных чисел дважды выбирает числа от 1 до 6. Если сумма выпавших цифр меньше 7 и играющий ввел число меньшее 7, то он выигрывает сделанную ставку. Если сумма выпавших цифр больше 7 и играющий ввел число большее 7, он также выигрывает сделанную ставку. Если играющий угадал сумму цифр, он получает в четыре раза больше очков, чем сделанная ставка. Ставка проиграна, если не имеет место ни одна из описанных ситуаций. В начальный момент у играющего 100 очков. Предусмотреть сохранение статистики результатов игроков.

Приложение 4

Образец пояснительной записки к курсовой работе на тему: «Программа для генерации кроссвордов из имеющих слов и описаний»

РЕФЕРАТ

Курсовая работа содержит пояснительную записку на 24 листах формата А4, 5 рисунков, 2 блок-схемы, 8 литературных источников.

В процессе работы была создана программа, позволяющая генерировать кроссворды по заданным исходным данным, а также закреплены знания программирования на языке С++.

В пояснительной записке описаны принципы создания кроссвордов, алгоритмы формирования кроссвордов из заданных слов и маски, реализовано краткое описания функций, которые будут использоваться для написания программы, сведения о самом языке программирования С++ и среде Microsoft Visual studio 2022.

Ключевые слова:

СЛОВО, ЦИКЛ, ФАЙЛ, КРОССВОРД, МАСКА, ШАБЛОН, ФУНКЦИЯ, ПЕРЕМЕННАЯ, АЛГОРИТМ, ПРИКЛАДНЫЕ ПРОГРАММЫ

ВВЕДЕНИЕ

Кроссворды стали популярны в начале XX века. Есть несколько версий того, где и когда они были изобретены, возможно, это произошло ещё в XIX веке. На звание родины кроссвордов претендуют Великобритания, США и ЮАР. По некоторым данным, первый кроссворд был напечатан в США в журнале «Св. Николас» в 1875 году. Первый кроссворд на русском языке составлен писателем В.В. Набоковым и опубликован в приложении «Наш мир» к газете «Руль» в феврале 1925 года в Берлине. Считается, что в России впервые кроссворд появился в журнале «Огонёк» в 1929 году [1].

В настоящее время существуют следующие разновидности кроссвордов: Сканворд, Линейный сканворд, Круговой сканворд, Одни согласные, Слоговый сканворд, Антисканворд, Антикроссворд, Японский кроссворд, Венгерский кроссворд, Американский кроссворд, Эстонский кроссворд, Антиэстонский кроссворд, Ключворд, Звёздочка, Октогон, Гексагон, Дуаль, Мозайка, Филворд, Числовой филворд, Криптоман, Числовой кроссворд, Числовой антикроссворд, Какуро, Анаграммное домино, Анаграмный кроссворд, Коврик, Улитка, Чайнворд, Диагональ, Эрудит-лото.

Для реализации поставленной задачи был использован язык программирования C++. C++ – это универсальный объектно-ориентированный язык программирования. В процессе разработки использовался пакет Microsoft Visual Studio, а именно Microsoft Visual C++ (MSVC) – интегрированная среда разработки приложений на языке C++, разработанная фирмой Microsoft и поставляемая как часть комплекта Microsoft Visual Studio. Эта среда обладает всеми возможностями для создания различных программ – от простых калькуляторов до программ, управляющих космическими спутниками.

Целью данной курсовой работы было написание программы по составлению кроссвордов по шаблону и словарю, это было достигнуто с помощью программного пакета Microsoft Visual Studio 2022 и языка C++.

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Постановка задачи

Целью данной курсовой работы является создание программы «Генерация кроссвордов».

Прежде всего рассмотрим, как же обычно люди составляют кроссворды.

Сначала человек придумывает или выбирает сетку, руководствуясь некими внешними соображениями, например, требованиями заказчика или эстетическими соображениями. Затем начинает, обычно с самого длинного слова, заполнять клетки сетки кроссворда. Если на некотором этапе подобрать очередное слово не представляется возможным, составитель пытается вернуться на шаг назад и подобрать другие слова.

Конечно, это очень грубый алгоритм действий человека. Человек подбирает слова для кроссворда, руководствуясь тематикой кроссворда, уровнем сложности, интересностью определений. Кроме того, человек сразу отсекает часть слов, осознанно или неосознанно проводя своего рода лексический анализ. Он учитывает, что некоторые сочетания букв в языке встречаются значительно реже, а некоторые вообще не встречаются, как например «Ь» в начале слова в русском языке.

Но, в целом, все эти тонкости, которые отличают хороший кроссворд от просто кроссворда, не являются препятствием для построения вполне применимой программы, генерирующей кроссворды.

Разработанная программа позволяет быстро и легко создать линейный кроссворд любой степени сложности, применяя готовые словари и шаблоны для генерации кроссворда.

Согласно поставленному заданию курсовой работы, при написании программы необходимо по заданной сетке составить кроссворд, используя слова из некоторого словаря. Другими словами, необходимо отобрать группу элементов (слов) из некоторого множества (словаря), каждый из которых должен удовлетворять некоторому уникальному признаку (маске).

Принимаются следующие ограничения.

1. Все сетки – квадратные, размерами n×n

2. Длины масок для слов находятся в пределах от 3 до 12.

3. Нумерация слов кроссворда производиться в порядке убывания длин их масок.

4. Слова кроссворда не могут начинаться с букв «Й, Ь, Ы, Ъ».

Хотя такое допущение и не является, вообще говоря, правомерным, так как мы заранее исключаем такие слова, как «йод» или «Ыгыата» из кроссворда, однако тем самым, в некоторых случаях мы существенно сократим время работы алгоритма.

Алгоритм не находит все возможные варианты кроссвордов, но строит некоторый случайным образом. Поэтому каждый новый запуск дает новый кроссворд.

Структуры данных, которые будут использоваться в работе:

Словарь

Словарь разбит на разделы. Каждый раздел содержит в себе слова одинаковой длины (от 3-х до 12-и букв). Словарь реализован как ассоциативный массив map<int, vector<string>>, в котором ключом служит длина слова. Таким образом, каждый раздел словаря представляет собой вектор строк, а доступ к разделу производится по значению длины слов, содержащихся в нем.

Сетка

Сетку кроссворда будем хранить в виде матрицы.

Маска

Маской будем называть последовательность, состоящую из пустых (свободных) клеток сетки кроссворда, предназначенных для записи в них слов, причем количество этих клеток должно быть от 3-х до 12-и.

Введем понятие релевантности слова маске.

Слово является релевантным маске в следующих случаях:

• оно имеет ту же длину, что и маска;

• соответственные (одинаковые по порядковому номеру) буквы слова совпадают с буквами, имеющимися в маске.

Реестр

Реестр – это вектор специальных структур данных, содержащий необходимую информацию обо всех масках будущего кроссворда.

Каждая строка реестра представляет информацию об одной маске.

Реестр должен хранить следующую информацию для каждой маски:

• индексы маски в матрице;

• адрес раздела словаря, содержащего слова той же длины, что и текущая маска;

• адрес слова в словаре (вписывается, когда релевантное слово найдено);

• длину маски;

• информацию о том, является данная маска вертикальной или горизонтальной.

Для представления реестра выделен отдельный файл, в котором хранятся длина, индексы первой клетки этой маски в матрице (x и y) и направление этой маски в матрице: по вертикали или по горизонтали.

1.2 Эскизный проект

Прежде всего определимся как выглядит кроссворд. Кроссворд – это таблица, где в пустые клетки пишут слова по вертикали или по горизонтали, а клетки, где писать ничего ненужно, закрашены одним цветом смотреть рисунок 1.1. Кроссворд в журнале выглядит следующим образом:

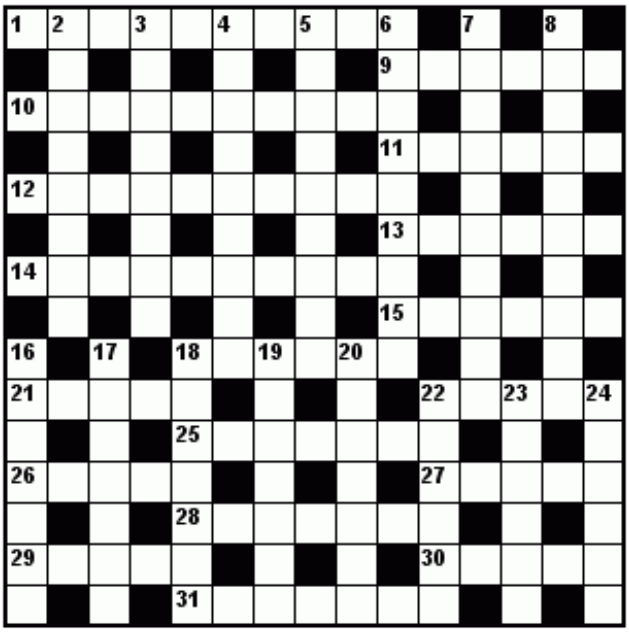


Рисунок 1.1 – Кроссворд в журналах и газетах

Графическая реализация в курсовой работе не преследовалась, т.к. курсовая работа предназначена для закрепления знаний по языку программирования C++, т.е. несет чисто учебный характер и рассчитана на обычного пользователя без каких-либо особых навыков работы за компьютером. Поэтому кроссворд в программе будет выглядеть как пересечения слов.

Программа должна быть проста в использовании, поэтому используется консольное приложение, а сам проект относится к Windows Console Application.

При ее запуске идет автоматическое подключение двух текстовых файлов. Первый «crossword.txt», в котором содержится реестр и маски, а именно первая строка длина и ширина кроссворда, дальнейшие строки – это маски слов, где первая цифра – x, вторая – y, третья длина слова, и потом метка горизонтального или вертикального расположения слова. Второй «vocabulary.txt», в котором содержатся слова, используемые при составлении кроссворда. Если кроссворд составить получится, то он выводится на экран, а если нет, то выводится сообщение о невозможности его генерации. Если есть

файл с вопросами и ответами к кроссворду, то он тоже будет выведен и называется «opis.txt». Все файлы предварительно заполняется вручную.

1.3 Алгоритм решения задачи

Этот раздел описывает алгоритм решения поставленной задачи.

Теперь составим последовательность действий, при выполнении которой, решается поставленная задача, а именно, пункты, описанные в предыдущем разделе.

Данная программа имеет следующий общий алгоритм.

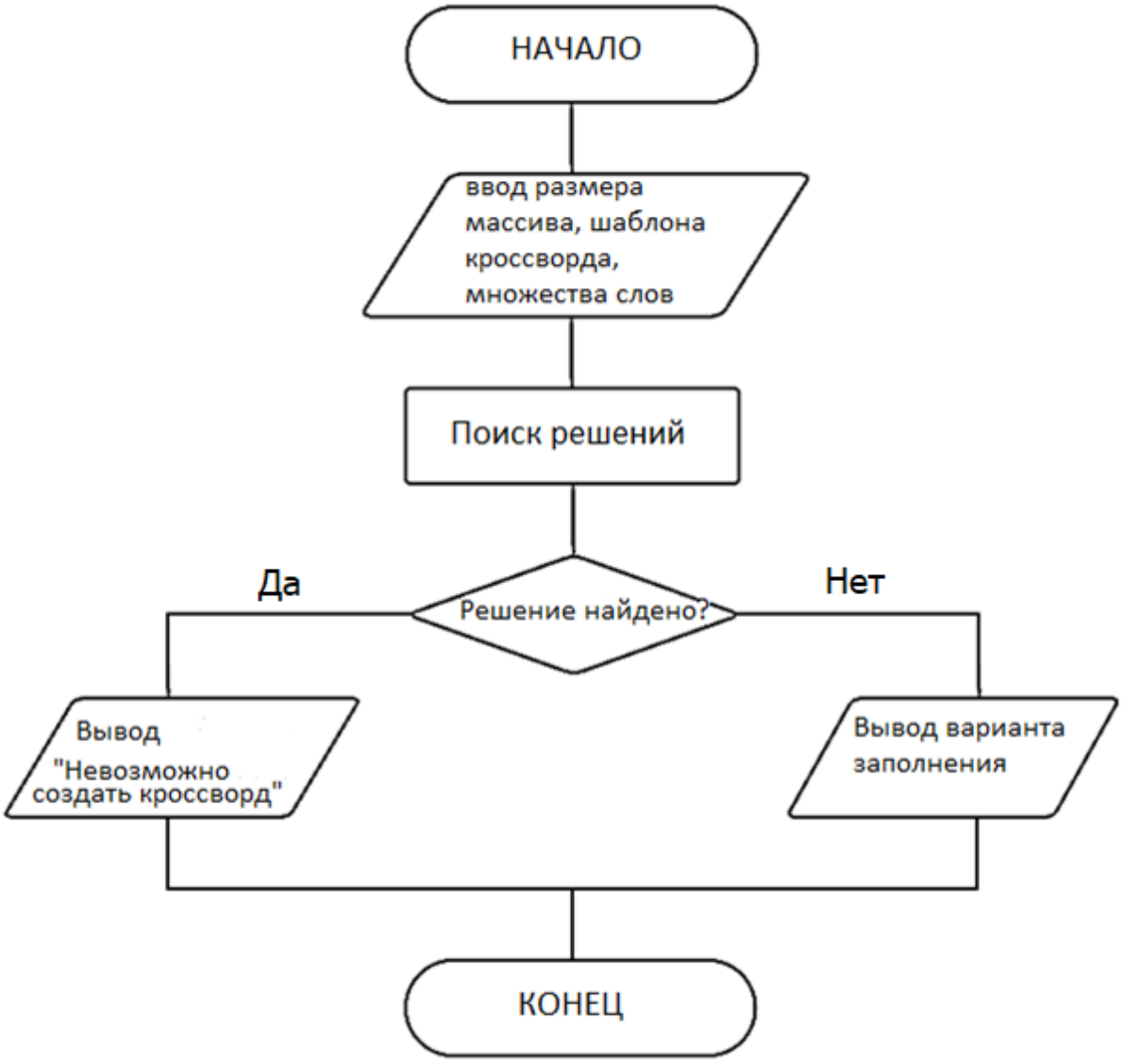


Рисунок 1.2 – Принципиальная блок-схема работы программы

Идея алгоритма сводится к следующему. Двигаясь по реестру сверху вниз от строки к строке, определяем адреса масок, выделяем и ищем в словаре релевантные им слова, которые затем вносим в матрицу. На каком-то этапе этот процесс может прерваться из-за отсутствия в словаре слова, релевантного рассматриваемой маске. Тогда по кросс шкале этой маски определяем слова, с которыми эта маска пересеклась, и только эти слова заменяем другими, релевантными своим маскам. В результате для неудачного слова возникнет другая маска, для которой в словаре, релевантное слово, возможно, найдется. При этом все ранее внесенные в матрицу слова, которые не пересекались с неудачной маской, остаются в матрице без изменения.

Другими словами, при неудаче на каком-то этапе работы по заполнению матрицы мы пытаемся «подправить» только те слова, которые помешали работе.

Описанная (в общей форме) процедура «подправления» работы по заполнению матрицы получила название коррекции.

Если коррекция не приводит к успеху, то следует выполнить новую попытку по повторному заполнению матрицы с самого начала.

Слово, релевантное какой-либо маске, разыскивается в словаре.

При выполнении этой процедуры используется информация, содержащаяся в реестре (точнее – в той строке реестра, которая относится к рассматриваемой маске). Работа по заполнению матрицы начинается с поиска слова, релевантного первой маске, то есть маске, характеризующейся первой строкой реестра. Если слово найдено, оно вносится в матрицу (в реестр при этом вносится адрес этого слова в словаре). После этого происходит переход к поиску второго слова, то есть к работе со второй строкой реестра [2].

1. Инициализация структур данных, описанных выше;
2. ЦИКЛ по реестру:
   1 поиск в словаре слова, релевантного заданной маске;
   2 ЕСЛИ слово найдено занесение слова в реестр,
   ИНАЧЕ выполнение коррекции.

Инициализация структур данных.

1. Считывание сетки из файла.

2. Заполнение словаря.

3. Построение реестра.

4. Заполнение кроссшкалы реестра.

После этого надо сделать копию матрицы, которая нам понадобится при коррекции для восстановления предыдущего состояния.

Поиск в словаре слова, релевантного заданной маске.

1. Из реестра получаем маску.

2. Из реестра получаем адрес начала соответствующего раздела словаря.

3. Случайным образом выбираем слово из данного раздела:

3.1. Если данное слово нерелевантно заданной маске, производится поиск по всему разделу словаря, начиная со следующего слова и до слова, стоящего перед выбранным случайно.

3.2. Если релевантное слово нашлось, проверяем, не было ли оно уже вписано в кроссворд ранее – ищем по предыдущим строкам реестра номер данного слова в поле, где хранится адрес слова в словаре.

3.3. Если данное слово уже встречалось, оно считается нерелевантным данной маске и поиск по словарю продолжается.

3.4. Если после просмотра всего раздела не нашлось ни одного релевантного слова, выполняется коррекция.

Коррекция.

Алгоритм построения массива корректирующих масок.

Выполнение коррекции сводится в основном к построению массива корректмасок.

В описании алгоритма использованы следующие обозначения [2]:

i номер строки реестра, на которой возникла неудача;

Ші – кроссшкала ій строки реестра;

j1, j2, j3 ... – числа, встречающиеся в Ш1,

Шj кроссшкалы, соответствующие j1, j2, j3 ... строкам реестра;

i1, i2, i3 ... числа, встречающиеся в Шj;

Шj «прореженная» кроссшкала jй строки реестра;

Fj корректмаска для jго слова матрицы.

Алгоритм построения корректмасок задается в виде следующей последовательности действий:

1. в заполняемой матрице восстанавливается исходное состояние маски i-го слова. Для этого i-я маска копируется из дубликата матрицы.

2. из кроссшкалы Шi выделяются все числа j, меньшие числа i: j1, j2, ...;

3. в кроссшкале Шj1, все числа 1, большие или равные числу i, заменяются на «пробел», то есть производится «прореживание».

Получаем «прореженную» кроссшкалу Шj1;

4. операции по п. (3) повторяем для остальных чисел j. Получаем прореженные кроссшкалы: Шj1, Шj2...;

5. в j1-м слове, выделенном из матрицы, все буквы заменяем на «пробелы», кроме тех, которым соответствуют какиелибо числа в Шj1 Символы «*» в j1-м слове сохраняются. Получаем коррект маску для j1-го слова Fj1;

6. операции по п. (5) повторяем для остальных чисел j. Получаем корректмаски: Fj1, Fj2, ...

Данный алгоритм, по сути, осуществляет в случае неуспеха возврат на один шаг назад.

После образования массива корректирующих масок следует выполнить поиск по полученным маскам соответствующих корректирующих слов и внести их в матрицу с тем, чтобы, вернувшись к ранее неуспешной маске (то есть к i-й строке реестра), попытаться найти для этой (теперь уже измененной) маски релевантное слово.

Если релевантного слова для i-й маски опять не найдется, то следует повторить поиск корректслов по тем же корректмаскам.

Случайный характер обращения к словарю приведет к нахождению новых корректслов, что в свою очередь приведет к возникновению новой маски для i-го слова, для которой релевантное слово, возможно, найдется. Поиск новых

корректслов следует повторять до достижения удачи (то есть до получения такой маски для i-го слова, для которой в словаре найдется релевантное слово).

Если же за некоторое предварительно определенное число N итераций, решения найти не удалось, поиск начинается с самого начала.

Однако такой алгоритм не всегда приводит нас к решению. Обычно неудача возникает на последних шагах, когда подбираются слова, состоящее из 3-х или 4-х букв, которые пересекаются с предыдущими словами по два, а то и по три раза, то есть две или три буквы уже заполнены. В таких случаях очень редко можно подобрать подходящие слова, даже после выполнения коррекции, так как пересекающиеся с неудачной маской слова сами пересекаются со словами, занесенными в матрицу еще раньше, и в итоге различных комбинаций оказывается не так много. Таким образом, для нахождения решения при таком алгоритме следует составлять сетки кроссворда так, чтобы было как можно меньше пересечений для коротких слов.

Рисунок 1.3 – Логическая схема алгоритма генерации линейных кроссвордов

2. ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1 Функции программы

Для реализации программы необходимо выделить следующие структуры [2].

Первая структура, описывающая отдельные буквы

```
struct CharAndCounter
{
	char Char;
	int Counter;
	CharAndCounter (char _char= ' ',int _counter=0) : Char (_char), Counter (_counter) {}
};
```

Вторая описывает отдельные слова словаря

```
	struct VocElement
{
	bool Busy;
 string String;
	VocElement (const string & str = "", bool b=false) : String (str), Busy (b) {}
};
```

Сам кроссворд описывается следующей структурой:

```
struct WordCoords
{
	int X,Y;
	char Dir;
	int Length;
WordCoords (int _x, int _y, int _len, char _dir) : X(_x), Y(_y), Dir(_dir), Length(_len) {}
	int dx() {return (Dir == HORIZONTAL) ? 1:0;}
	int dy() {return (Dir == VERTICAL) ? 1:0;}
};
```

Для считывания описания кроссворда из файла «crossword.txt» и словаря из файла «vocabulary.txt».

```
void ReadData()
{
	ifstream crossw("crossword.txt"), voc("vocabulary.txt");
	string temp;
if (crossw && voc)
{
	while (!voc.eof())
	{
voc>>temp;
Vocabulary.push_back(VocElement(temp,false));
	}
sort(Vocabulary.begin(),Vocabulary.end(),Less);
int W,H,x,y,len;
char dir;
```

```
crossw>>W;
crossw>>H;
for(;;)
{
        crossw>>x;
        crossw>>y;
        crossw>>len;
        crossw>>dir;
        if(crossw.eof())
 break;
        Crossword.push_back(WordCoords(x,y,len,dir));
}
for (int i=0;i<W;i++)
{
        vector<CharAndCounter> col(H);
        fill(col.begin(),col.end(),CharAndCounter());
        Field.push_back(col);
}
}
else
cout<<"Ошибка подключения словаря и шаблона"<<endl;
}
```

Для сортировки слов по длине напишем функцию Less.

```
bool Less (const VocElement& lhs, const VocElement & rhs)
{
        return lhs.String.length() < rhs.String.length();
}
```

Функция для проверки вписания слова в определенную позицию, при условии, что длина слова устраивает.

```
bool CanPlace(WordCoords c,const string& word)
{
 for (unsigned i=0;i<word.length();i++)
 {
if (Field[c.X+i*c.dx()][c.Y+i*c.dy()].Char != ' ' && Field[c.X+i*c.dx()][c.Y+i*c.dy()].Char !=word [i])
        return false;
 }
 return true;
}
```

Функция размещения слова, предполагается, что это возможно сделать.

```
void PlaceWord(WordCoords c, const string& word)
{
        for (unsigned i=0;i<word.length();i++)
        {
                Field[c.X+i*c.dx()][c.Y+i*c.dy()].Char = word[i];
                Field[c.X+i*c.dx()][c.Y+i*c.dy()].Counter++;
        }
}
```

Функция снятия слова с позиции, на которую хотели поместить слово.

```
void RemoveWord(WordCoords c, const string& word)
```

```
{
for (unsigned i=0;i<word.length();i++)
        {
                if (--Field[c.X+i*c.dx()][c.Y+i*c.dy()].Counter ==0)
                Field[c.X+i*c.dx()][c.Y+i*c.dy()].Char=' ';
        }
}
```

Основная функция генерации кроссворда. Эта функция перебирает все незанятые слова словаря.

```
bool Solve(unsigned CoordNo)
{
if (CoordNo == Crossword.size())
return true;
pair <vector<VocElement>::iterator, vector<VocElement>::iterator> range=
 equal_range (Vocabulary.begin(),Vocabulary.end(),string(Crossword[CoordNo].Length,' '),Less);
for (vector<VocElement>::iterator p=range.first; p!= range.second;p++)
if(!p->Busy && CanPlace(Crossword[CoordNo],p->String))
{
        PlaceWord(Crossword[CoordNo],p->String);
        p->Busy=true;
        if (Solve(CoordNo+1))
                return true;
        RemoveWord(Crossword[CoordNo],p->String);
        p->Busy=false;
}
return false;
}
```

2.2 Интерфейс и тестирование программы

В результате работы был получен exe-файл, который запускается двойным кликом мыши. При запуске программы появляется следующее консольное окно, показывающее работу программы. Смотрите рисунок 2.1.

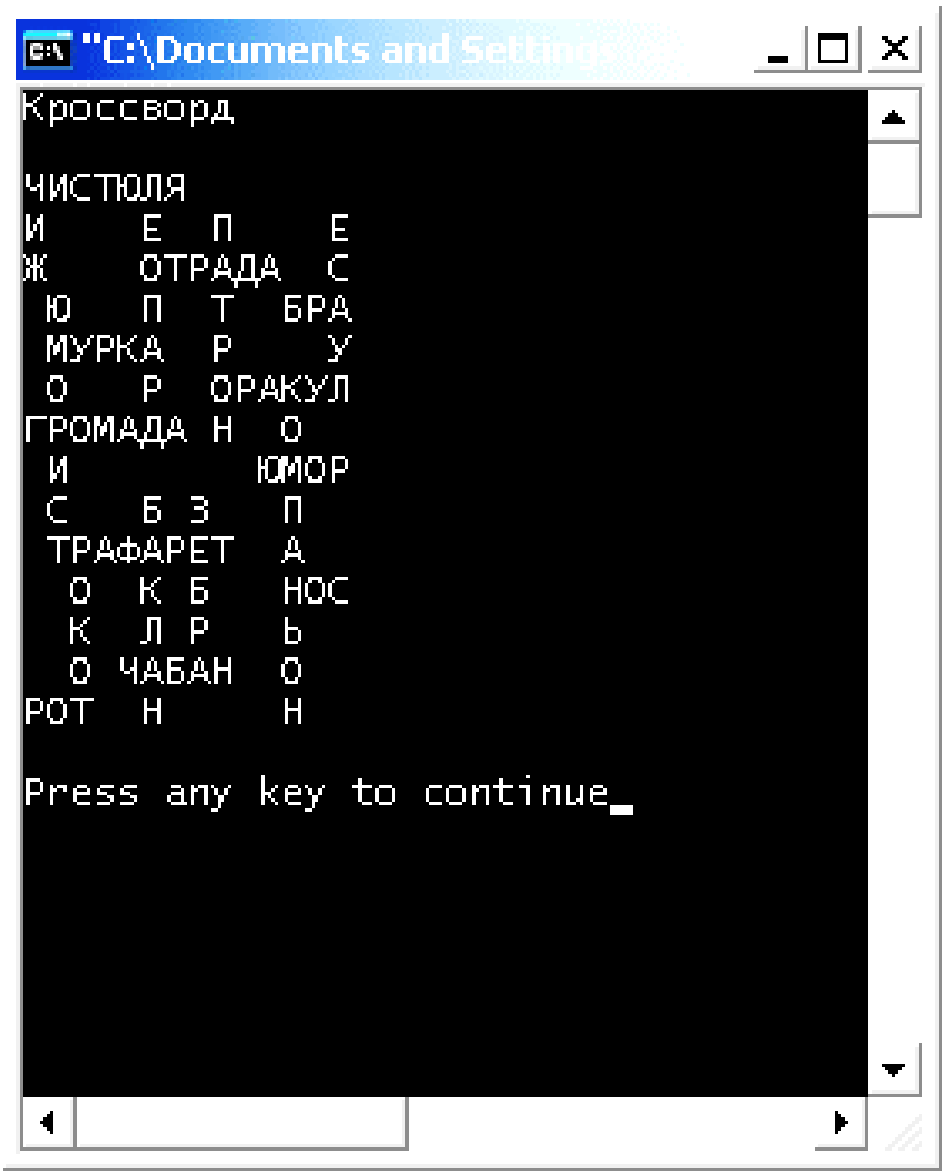


Рисунок 2.1 – Генерация кроссворда – пример

Если создать кроссворд не получится или неверное имя файлов, используемых для создания кроссворда, то будет выведено сообщение о невозможности работы. Результат такого запуска показан на рисунке 2.2 и рисунке 2.3.

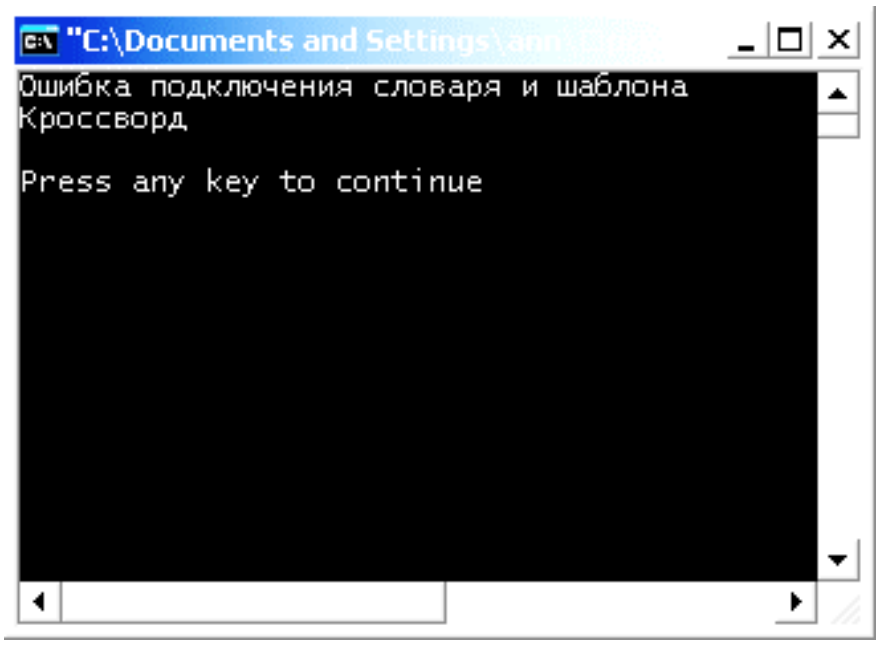


Рисунок 2.2 – Ошибка подключения файлов шаблона или словаря

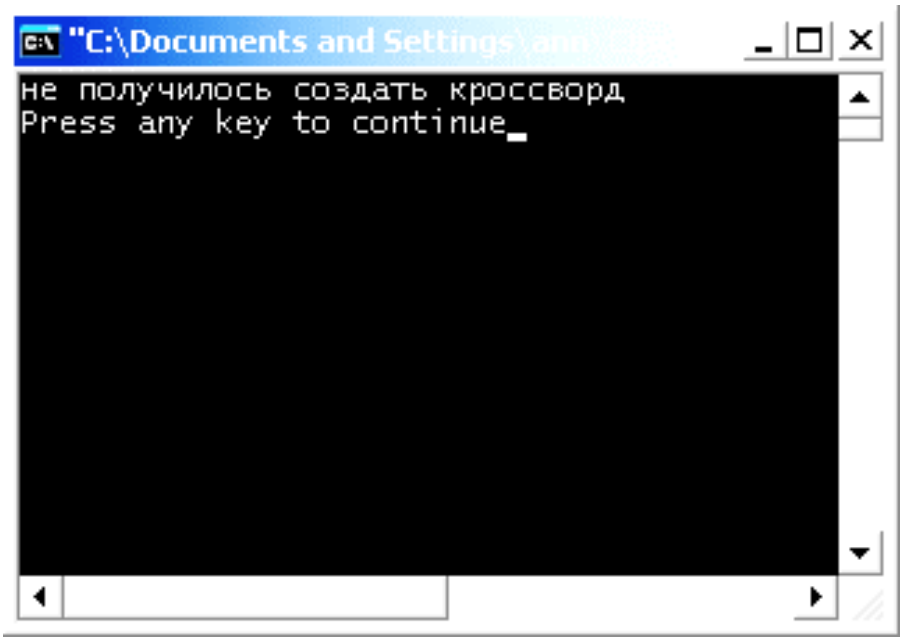


Рисунок 2.3 – Ошибка составления кроссворда

Если к кроссворду составить описание и записать его в файл, то будет выведен следующий результат. Смотрите рисунок 2.4.

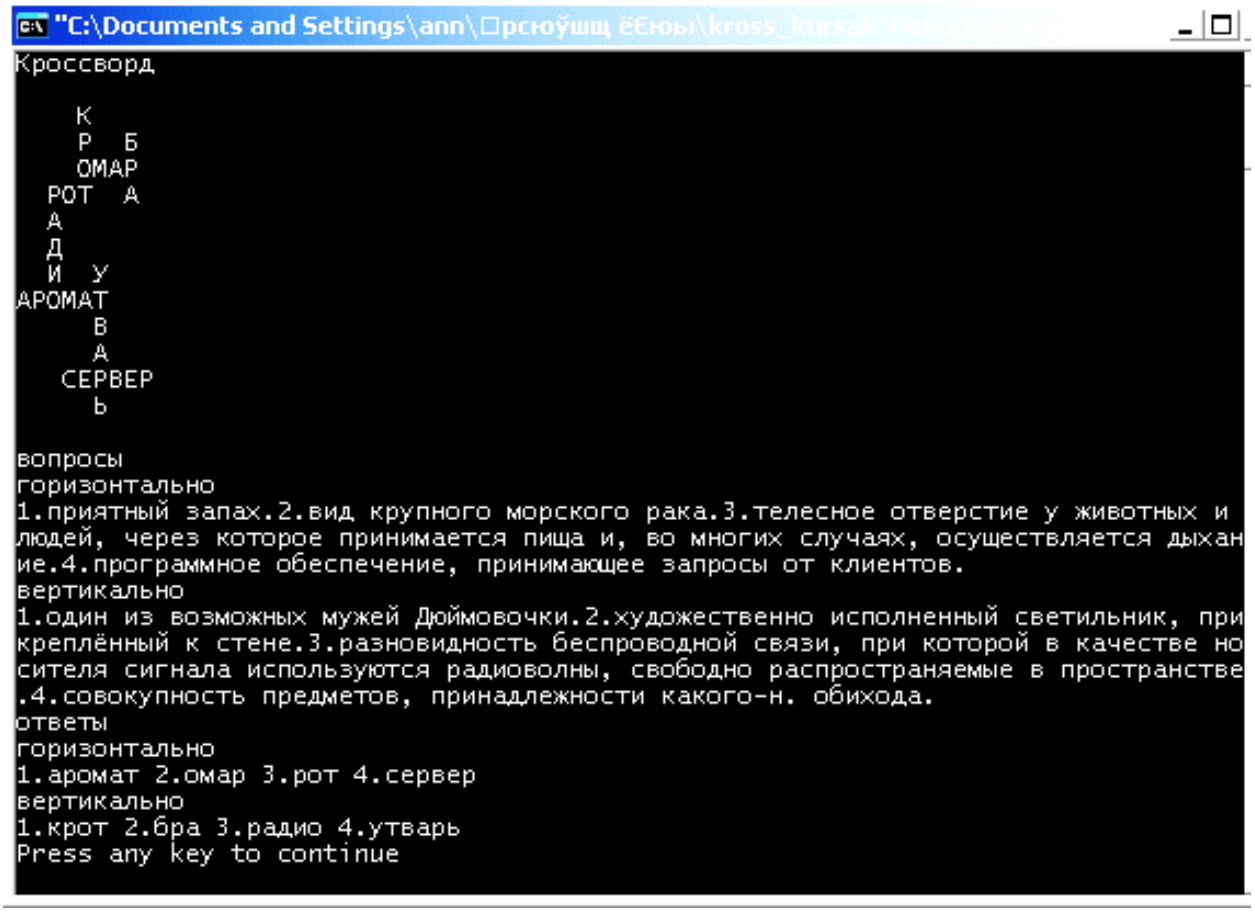


Рисунок 2.4 – Кроссворд с описанием

ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы была разработана программа, которая генерирует кроссворд из предоставленного словаря-файла и шаблона-файла. Программа имеет консольный интерфейс, так как цель работы заключалась в разработке алгоритмов и изучении методов составления кроссвордов, а полученные функции в дальнейшем могут быть использованы как модуль в больших игровых программах.

Во время написания данной работы были изучены способы работы с функциями, указателями, классами, файлами и различными библиотеками C++.

Приложение 5

Образец списка используемой литературы

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. История кроссворда – [электронный ресурс]. – Режим доступа : https://krosswordist.ru/difnews/9-istoriya-krossvorda.html. – Загл с экрана. – Дата обращения: 20.12.2024.

2. Мозговой, М. В. C++ Мастер-класс. 85 нетривиальных проектов, решений и задач / М. В. Мозговой. – СПб. : Наука и техника, 2007. – 272 с.

3. Павловская, Т. А. C/C++. Программирование на языке высокого уровня / Т. А. Павловская. – СПб. : Питер, 2021. – 461 с.

4. Прата, Стивен. Язык программирования C++. Лекции и упражнения / Стивен Прата. – М. : ООО «И.Д. Вильямс», 2018. – 1244 с.

5. Доусон, М. Изучаем C++ через программирование игр / Майкл Доусон. – СПб. : Питер, 2016. – 352 с.

6. Скотт, Мейерс Эффективный и современный C++ : 42 рекомендации по использованию C++11 и C++14 / Скотт Мейерс. : пер. с англ. – М. : ООО «И.Д. Вильямс», 2016. – 304 с.

7. Липпман, С. Язык программирования C++. Полное руководство / С. Липпман, Ж. Лажойе; перевод А. Слинкин. – 3-е изд. – Саратов : Профобразование, 2024. – 1104 c. – ISBN 978-5-4488-0136-5. – Текст : электронный // Цифровой образовательный ресурс IPR SMART: [сайт]. – URL: https://www.iprbookshop.ru/145938.html (дата обращения: 27.12.2024). – Режим доступа: для авторизир. пользователей

8. Ламберт, Д. Microsoft Word 2016 / Д. Ламберт – М. : Эком, 2019. – 656 с.

9. Леонтьев, В. П. Microsoft Word 2016 : новейший самоучитель / Виталий Леонтьев. – М. : «Э», 2016. – 128 с.

Учебное издание

ТИВОНЕНКО АННА АЛЕКСАНДРОВНА
ХИТРЫХ ОЛЬГА ВЛАДИМИРОВНА

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ
по выполнению курсовой работы

Методические рекомендации

В авторской редакции

Подписано в печать 25.06.2025. Бумага офсетная. Гарнитура Times New Roman.
Печать ризографическая. Формат 60×84/16. Усл. печ. л. 2,44.
Тираж 100 экз. Зак. № 52.

ФГБОУ ВО «ЛГПУ»
Издательство ЛГПУ
ул. Оборонная, 2, г. Луганск, ЛНР, 291011. Т/ф: +7-857-2-58-03-20.
e-mail: knitaizd@mail.ru