

О.В. ПЛАТОНОВА, Ю.С. АСАДОВА, М.М. РАСУЛОВ

**АЛГОРИТМИЧЕСКИЕ ОСНОВЫ ОБРАБОТКИ
ДАННЫХ**

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Москва — 2022

УДК 004
ББК 32.972
П __

Платонова О.В. Алгоритмические основы обработки данных [Электронный ресурс]: Методические указания / Платонова О.В., Асадова Ю.С., Расулов М.М. — М.: МИРЭА – Российский технологический университет, 2022. — 1 электрон. опт. диск (CD-ROM)

Разработаны в помощь студентам, выполняющим курсовую работу по дисциплине «Алгоритмические основы обработки данных». В методических указаниях приведены варианты задания и общие методические указания для выполнения курсовой работы.

Предназначено для студентов, обучающихся по направлению подготовки бакалавров 09.03.01 «Информатика и вычислительная техника».

Методические указания издаются в авторской редакции.

Авторский коллектив: Платонова Ольга Владимировна, Асадова Юлия Сергеевна, Расулов Мирзо Максудович

Рецензенты:

Андрианова Елена Гельевна, к.т.н., доцент, заведующая кафедрой Корпоративных информационных систем, РТУ МИРЭА.

Минимальные системные требования:

Наличие операционной системы Windows, поддерживаемой производителем.

Наличие свободного места в оперативной памяти не менее 128 Мб.

Наличие свободного места в памяти хранения (на жестком диске) не менее 30 Мб.

Наличие интерфейса ввода информации.

Дополнительные программные средства: программа для чтения pdf-файлов (Adobe Reader).

Подписано к использованию по решению Редакционно-издательского совета

МИРЭА – Российского технологического университета от _____ 2022 г.

Объем ____ Мб

Тираж 10

© Платонова О.В., Асадова Ю.С., Расулов М.М., 2022

© МИРЭА – Российский технологический университет, 2022

СОДЕРЖАНИЕ

Введение	4
1. Руководство выполнением курсовой работы	6
2. Порядок выполнения курсовой работы	7
3. Темы курсовой работы.....	8
4. Структура и содержание курсовой работы.....	21
5. Основная часть курсовой работы	23
5.1. Техническое задание	23
5.2. Обзор способов организации данных и обоснование выбора структуры данных для эффективного выполнения операций	24
5.3. Описание программы	24
6. Требования по оформлению курсовой работы	25
6.1. Общие положения	25
6.2. Построение текста	27
6.2.1. Разделы, подразделы, пункты	27
6.2.2. Заголовки.....	28
6.3. Содержание	31
6.4. Маркированные и нумерованные списки	31
6.5. Формулы, уравнения и переменные	32
6.6. Таблицы	34
6.7. Иллюстрации	36
6.8 Текст (листинг) программ	38
6.9. Список использованных источников	39
6.10. Приложения	41
7. Защита курсовой работы	43
Список использованных источников	45
Приложение 1	46
Приложение 2	47
Приложение 3	48
Приложение 4	49
Приложение 5	51
Приложение 6	52
Приложение 7	55
Приложение 8	58
Приложение 9	73
Сведения об авторах.....	74

ВВЕДЕНИЕ

Методические указания разработаны на основе Инструкции по организации и проведению курсового проектирования в Федеральном государственном бюджетном образовательном учреждении высшего образования «МИРЭА – Российский технологический университет». СМКО МИРЭА 7.5.1/04.И.05-18 [1]. Данной инструкцией определяется понятие курсовой работы, требования к ее объему, структуре и оформлению, раскрывается порядок выполнения и защиты.

В современных условиях становления и развития цифровой экономики выпускники вуза должны быть подготовленными к самостоятельной профессиональной деятельности. Важной формой развития навыков самостоятельной работы является курсовое проектирование.

Курсовая работа (КР) является формой самостоятельной работы обучающихся под руководством преподавателя и представляет собой творческую, самостоятельную работу обучающихся, имеющую целью формирование у них компетенций в соответствии с матрицей компетенций ООП и требованиями ФГОС ВО. КР выполняются в объёме, отведённом учебным планом на изучение дисциплины «Алгоритмические основы обработки данных».

Основной целью КР является формирование и закрепление компетенций путём практического использования знаний, умений и навыков, полученных в рамках теоретического обучения, а также выработка самостоятельного творческого подхода к решению профессиональных задач.

КР представляет собой научно-методическую письменную работу, целью которой является развитие творческих навыков, в том числе в области научных исследований, и детальное изучение структур данных и алгоритмов их обработки, а также получение практических навыков их использования при разработке программ.

КР выполняется лично студентом под руководством преподавателя, в задании на курсовую работу для каждого обучающегося устанавливается конкретная задача и ожидаемые результаты.

Руководителем КР, является преподаватель, ведущий данную дисциплину.

Обучающийся имеет право выбора темы КР из списка, предложенного в соответствующем разделе настоящих методических указаний, которое оформляется личным заявлением на имя заведующего кафедрой. Обучающийся может

предложить свою тему при условии обоснования ее целесообразности. Темы КР обучающихся должны быть определены не позднее трех недель с начала соответствующего семестра.

Закрепление тем КР за обучающимися и назначение руководителей производится распоряжением заведующего кафедрой.

КР (при условии успешной защиты) является одной из форм отчетности студента по итогам обучения за соответствующий семестр, свидетельствующей о выполнении учебного плана.

Требования к структуре и содержанию КР определяются в настоящих методических указаниях.

1. РУКОВОДСТВО ВЫПОЛНЕНИЕМ КУРСОВОЙ РАБОТЫ

Научный руководитель КР определяется в соответствии с утвержденной нагрузкой на текущий учебный год.

Научный руководитель выполняет следующие функции:

- разрабатывает темы задания на КР в соответствии с решением кафедры о закреплении тем КР;
- согласовывает с обучающимся тему работы;
- устанавливает (конкретизирует) требования к содержанию и объему КР на основе настоящих методических указаний и доводит их до сведения студентов при выдаче заданий на КР;
- определяет основные направления деятельности обучающихся по выполнению КР в соответствии с заданиями;
- рекомендует научную литературу и другие источники информации по выбранной теме;
- осуществляет контроль за процессом курсового проектирования и консультирование обучающегося по вопросам выполнения КР в соответствии с расписанием, утверждаемым заведующим кафедрой, которое вывешивается на информационном стенде кафедры или/и представляется на сайте кафедры не позднее срока утверждения тем КР, или по составленному совместно с обучающимся графику индивидуальных консультаций;
- осуществляет контроль за выполнением КР;
- оценивает содержание КР;
- подготавливает отзыв на КР;
- допускает к защите КР.

2. ПОРЯДОК ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

Выполнение курсовой работы по дисциплине «Алгоритмические основы обработки данных» включает в себя следующие этапы:

1. Выбор темы КР.
2. Заполнение бланка задания на КР (Приложение 2).
3. Подбор источников информации по теме КР.
4. Составление плана КР.
5. Систематизация и логическое изложение материала в соответствии с планом работы.
6. Заключение (выводы).
7. Оформление КР.
8. Получение отзыва научного руководителя на КР (Приложение 3).
9. Получение допуска к защите КР от научного руководителя.
10. Защита КР.

3. ТЕМЫ КУРСОВОЙ РАБОТЫ

Студент может выбрать тему курсовой работы из приведенного ниже списка или предложить свою тему, согласовав ее с преподавателем.

1. «Словарь с дубликатами»

Словарь с дубликатами – это структура данных, обеспечивающая доступ к информации по ключу и допускающая хранение элементов данных с одинаковыми ключами. Элемент словаря состоит из ключа и связанного с ключом значения. Пример словаря с дубликатами: англо-русский словарь, элементы которого содержат слово на английском языке (ключ) и вариант перевода слова на русский язык (значение), например, {[list] – список, [list] – перечислять}.

Операции словаря:

- создание пустого словаря;
- добавление элемента в словарь;
- исключение элементов с заданным ключом из словаря;
- поиск значений по ключу;
- изменение значения элемента словаря;
- вывод словаря в порядке возрастания ключей.

В демонстрационной программе предусмотреть считывание словаря в начале работы программы из текстового файла и запись словаря в файл перед завершением ее работы.

2. «Предметный указатель»

Предметный указатель – это алфавитный список терминов документа с указанием страниц, на которых они упоминаются. Пример элемента указателя: {[заголовочный файл], 108, 189, 927}.

Операции указателя:

- создание пустого предметного указателя;
- добавление термина в предметный указатель;
- редактирование элемента предметного указателя (добавление ссылки на страницу для заданного термина);
- вывод элементов указателя в алфавитном порядке на экран;
- вывод элементов указателя в алфавитном порядке в текстовый файл;
- поиск элемента предметного указателя по термину.

В демонстрационной программе предусмотреть ввод списка терминов с клавиатуры, считывание текста, для которого создается предметный указатель, из текстового файла и запись предметного указателя в другой текстовый файл перед завершением работы программы. Количество строк на странице текста

можно задать глобальной константой или вводить с клавиатуры. В текстовом файле нет переноса слов.

3. «Множество»

Множество – это совокупность элементов с уникальными значениями, которые, как правило, имеют одинаковый тип данных. Множество должно быть реализовано на основе динамического массива. Реализовать множество таким образом, чтобы можно было легко изменить тип его элементов (использовать шаблоны функций).

Операции множества:

- создание пустого множества;
- включение элемента в множество;
- исключение элемента из множества;
- проверка вхождения элемента в множество;
- объединение двух множеств;
- пересечение двух множеств;
- разность двух множеств;
- проверка равенства двух множеств.

В демонстрационной программе предусмотреть считывание множеств в начале работы программы из текстовых файлов и запись множеств в файлы перед завершением работы программы.

4. «Словарь»

Словарь – это индексируемая структура данных, доступ к элементам которой выполняется только по индексу (ключу). Элемент словаря состоит из ключа и связанного с ключом значения. Пример словаря: русско-английский словарь, элементы которого содержат слово на русском языке (ключ) и перевод слова на английский язык, например, {[список], list}. Для хранения словаря в оперативной памяти использовать динамический массив.

Операции словаря:

- создание пустого словаря;
- добавление элемента в словарь;
- исключение элемента из словаря;
- поиск элемента словаря по ключу;
- изменение значения элемента;
- вывод словаря в порядке возрастания ключей.

В демонстрационной программе предусмотреть считывание словаря в начале работы программы из текстового файла и запись словаря в файл перед

завершением работы программы. Элементом словаря является абонент телефонной сети (номер телефона, ФИО, паспортные данные). Поле «номер телефона» является ключом. ФИО и паспортные данные являются значением элемента словаря.

5. «Полином»

Полином n -степени задается формулой:

$$a_0x^n + a_1x^{n-1} + a_2x^{n-2} + \dots + a_n$$

Коэффициенты полинома должны храниться в динамическом массиве.

Операции:

- ввод коэффициентов полинома из текстового файла (первое число файла – степень полинома);
- вывод полинома на экран в виде: $4x^3 + 5x^2 - 2x^1 - 6$;
- умножение полинома на число;
- изменение знаков коэффициентов полинома на противоположные;
- сложение двух полиномов;
- произведение двух полиномов;
- присваивание одного полинома другому;
- проверка равенства двух полиномов;
- запись полинома в файл: первое число файла – степень полинома, следующие числа - его коэффициенты.

В демонстрационной программе предусмотреть запись выполняемых операций с полиномами в текстовые файлы. Форматы вывода бинарной и унарной операций:

(полином_1) знак_бинарной операции (полином_2) -> результат

знак_унарной операции (полином) -> результат

6. «Нижняя треугольная матрица чисел»

В нижней треугольной матрице элементы, находящиеся выше главной диагонали, равны нулю.

Для хранения значений матрицы для экономии памяти используется одномерный динамический массив, в котором хранятся только элементы из нижней области матрицы. Пример хранения матрицы A из четырех строк в одномерном массиве B приведен на рис. 3.1.

В демонстрационной программе предусмотреть запись матриц в файлы перед завершением работы программы.

Операции:

- создание нижней треугольной матрицы заданного размера с нулевыми

значениями всех элементов;

- установка значения элемента матрицы с индексами i, j ;
- получение значения элемента матрицы с индексами i, j ;
- построчный ввод матрицы;
- построчный вывод матрицы;
- сложение двух треугольных матриц;
- проверка равенства двух матриц;
- запись матрицы в файл (первое число файла – размер матрицы);
- чтение матрицы из текстового файла (первое число файла – размер матрицы);
- присваивание одной матрицы другой.

Матрица A	Массив B
A ₀₀ 0 0 0	
A ₁₀ A ₁₁ 0 0	B ₀ B ₁ B ₂ B ₃ B ₄ B ₅ B ₆ B ₇ B ₈ B ₉
A ₂₀ A ₂₁ A ₂₂ 0	A ₀₀ A ₁₀ A ₁₁ A ₂₀ A ₂₁ A ₂₂ A ₃₀ A ₃₁ A ₃₂ A ₃₃
A ₃₀ A ₃₁ A ₃₂ A ₃₃	Значения элементов массива B

Рисунок 3.1. Хранение нижней треугольной матрицы в одномерном массиве

7. «Конкурс»

Программа должна содержать функции для проведения конкурса. Количество участников конкурса и количество призовых мест, на которые претендуют участники, ограничено. Участник конкурса имеет идентификационный номер, имя. Во время проведения конкурса его участникам присваивают баллы.

Операции конкурса:

- открытие конкурса;
- регистрация участника;
- окончание регистрации участников;
- проведение конкурса: регистрация балла, полученного участником конкурса;
- окончание конкурса;
- подведение итогов конкурса: сортировка списка участников по убыванию баллов и вывод упорядоченного списка участников на экран и в текстовый файл.

Регистрация участников может происходить в течение нескольких сеансов работы программы, поэтому при завершении работы программы (до окончания регистрации участников) необходимо участников, зарегистрированных в теку-

щем сеансе, дописывать в конец текстового файла. Регистрация результатов конкурса и подведение итогов выполняется в течение одного сеанса работы программы.

8. «Числовая матрица с изменяемыми размерами»

Матрица должна быть представлена в памяти в виде указателя на массив указателей на одномерные динамические массивы чисел.

Операции:

- создание матрицы из данных текстового файла;
- установка значения элемента с заданными индексами (если индексы выходят за границы размеров матрицы, то должно выдаваться сообщение об ошибке);
- получение значения элемента матрицы с заданными индексами (если индексы выходят за границы размеров матрицы, то должно выдаваться сообщение об ошибке);
- изменение количества строк матрицы (если при этом матрица увеличивается, то новые элементы заполняются нулевыми значениями, а если уменьшается, то сохраняются только элементы из верхней левой части матрицы);
- изменение количества столбцов матрицы (если при этом матрица увеличивается, то новые элементы заполняются нулевыми значениями, а если уменьшается, то сохраняются только элементы из верхней левой части матрицы);
- получение текущего количества столбцов;
- получение текущего количества строк;
- вывод матрицы.

В демонстрационной программе предусмотреть считывание матрицы в начале работы программы из текстового файла и запись матрицы в файл перед завершением программы. Первые два числа в файле задают количество строк и столбцов матрицы.

9. «Множество целых чисел из диапазона 0-n»

Множество – это совокупность элементов с уникальными значениями, которые, как правило, имеют одинаковый тип данных.

Множество моделируется динамическим массивом из элементов, значения которых могут быть true или false. При включении числа со значением i в множество в элемент массива с индексом i записывается значение true, а при исключении числа со значением i из множества в элемент массива с индексом i записывается значение false.

Операции множества:

- создание пустого множества;
- включение элемента в множество;
- исключение элемента из множества;
- проверка вхождения элемента в множество;
- объединение двух множеств;
- пересечение двух множеств;
- разность двух множеств;
- проверка равенства двух множеств.

Пример множества $\{1, 3, 4\}$ приведен на рис. 3.2.

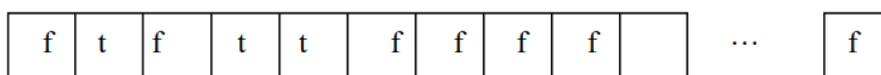


Рисунок 3.2. Хранение множества $\{1, 3, 4\}$ в массиве

В демонстрационной программе предусмотреть считывание множеств в начале работы программы из текстовых файлов и запись множеств в файлы перед завершением работы программы.

10. «Ежедневник»

Ежедневник должен содержать список ежедневных дел и отслеживать своевременность их выполнения.

Операции ежедневника:

- создание пустого ежедневника;
- добавление нового элемента в список дел;
- поиск элементов ежедневника по дате и вывод их, упорядоченных по времени выполнения;
- вывод всего списка элементов ежедневника, упорядоченного по сроку выполнения;
- исключение выполненного элемента из списка дел;
- очистка ежедневника.

В демонстрационной программе предусмотреть считывание существующего списка дел из текстового файла при каждом запуске программы и запись всех невыполненных дел в файл перед завершением работы программы.

11. «Множество целых чисел из диапазона $[0-n]$ с битовой схемой хранения»

Множество – это совокупность элементов с уникальными значениями, которые, как правило, имеют одинаковый тип данных.

Для множества используется массив целых чисел (4-х байтных). Каждый элемент множества хранится в виде 1 в соответствующем бите (с номером j)

элемента массива с индексом i , что уменьшает размер памяти необходимой множеству. Для выполнения операций с множествами с битовой схемой хранения используются быстрые поразрядные операции.

Операции множества:

- создание пустого множества;
- включение элемента в множество;
- исключение элемента из множества;
- проверка вхождения элемента в множество;
- объединение двух множеств;
- пересечение двух множеств;
- разность двух множеств;
- проверка равенства двух множеств;
- вычисление индекса элемента массива, в котором хранится элемент со значением x (вспомогательная операция);
- вычисление номера разряда элемента массива, в котором хранится элемент со значением x (вспомогательная операция).

В демонстрационной программе предусмотреть запись множеств в текстовые файлы перед завершением программы.

На рис. 3.3 приведена схема хранения множества $\{1, 30, 32, 64\}$.

Значение элементов	31	30	29...	2	1	0	63	62	...	33	32	95	94	...	65	64
Значения битов	0	1	0...	0	1	0	0	...	0	1	0	...	0	1	0	1
Номер разряда	31	30	29...	2	1	0	31	30	...	1	0	31	30	...	1	0
Элемент массива	a[0]						a[1]						a[2]			

Рисунок 3.3. Схема хранения множества $\{1, 30, 32, 64\}$

12. «Частотный словарь»

Частотный словарь – это алфавитный список всех слов заданного текста с указанием количества вхождений слова в текст.

Операции словаря:

- создание пустого словаря;
- добавление нового элемента в словарь;
- увеличение значения счетчика слова в элементе словаря с заданным словом;
- вывод элементов словаря в алфавитном порядке;
- поиск элемента словаря по слову.

В демонстрационной программе предусмотреть считывание текста для построения словаря из текстового файла и запись словаря в алфавитном порядке в другой текстовый файл перед завершением работы программы, считывание частотного словаря из текстового файла для просмотра и поиска в нем элементов. В текстовом файле не предусмотрено переноса слов.

13. «Средства ведения и обработки массива структур»

В массиве структур каждый элемент имеет признак «занят/свободен». Это позволяет при удалении элемента не перемещать в массиве элементы, а только изменять значение признака.

Операции:

- создание массива структур из данных текстового файла;
- изменение значения элемента с заданным (логическим) номером (логический номер структуры может не совпадать со значением индекса элемента, так как индекс учитывает все элементы массива, в том числе и свободные);
- получение значения элемента матрицы с заданным логическим номером;
- удаление элемента с заданным логическим номером;
- добавление элемента в массив (элемент добавляется в конец массива после последнего занятого элемента, а если последний занятый элемент является последним элементом массива, то перед добавлением нового элемента должно выполняться сжатие массива: исключение всех свободных элементов);
- получение текущего количества занятых элементов;
- получение текущего количества строк;
- вывод массива структур на экран;
- запись занятых элементов структур в текстовый файл.

Поля структуры: номер телефона, ФИО, адрес.

В демонстрационной программе предусмотреть считывание массива структур в начале работы программы из текстового файла и запись массива в файл перед завершением работы программы.

14. «Верхняя треугольная матрица чисел»

В верхней треугольной матрице элементы, находящиеся ниже главной диагонали, равны нулю.

Для хранения значений матрицы для экономии памяти используется одномерный динамический массив, в котором хранятся только элементы из верхней области матрицы. Пример хранения матрицы A из четырех строк в одномерном массиве B приведен на рис. 3.4.

Матрица A				Массив B									
A ₀₀	A ₀₁	A ₀₂	A ₀₃	B ₀	B ₁	B ₂	B ₃	B ₄	B ₅	B ₆	B ₇	B ₈	B ₉
0	A ₁₁	A ₁₂	A ₁₃	A ₀₀	A ₀₁	A ₀₂	A ₀₃	A ₁₁	A ₁₂	A ₁₃	A ₂₂	A ₂₃	A ₃₃
0	0	A ₂₂	A ₂₃	Значения элементов массива B									
0	0	0	A ₃₃										

Рисунок 3.4. Хранение верхней треугольной матрицы в одномерном массиве

Операции:

- создание верхней треугольной матрицы заданного размера с нулевыми значениями всех элементов;
- установка значения элемента матрицы с индексами i, j ;
- получение значения элемента матрицы с индексами i, j ;
- построчный ввод матрицы;
- построчный вывод на экран матрицы;
- разность двух треугольных матриц;
- проверка равенства двух матриц;
- запись матрицы в файл (первое число файла – размер матрицы);
- чтение матрицы из текстового файла (первое число файла – размер матрицы);
- присваивание одной матрицы другой.

В демонстрационной программе предусмотреть запись матриц в файлы перед завершением программы.

15. «Предметный указатель»

Предметный указатель – это алфавитный список терминов документа с указанием страниц, на которых они упоминаются. Пример элемента указателя: {[заголовочный файл], 108, 189, 927}.

Операции указателя:

- создание пустого предметного указателя;
- добавление термина в предметный указатель;
- редактирование элемента предметного указателя (добавление ссылки на страницу для заданного термина);
- вывод элементов указателя в алфавитном порядке на экран;
- вывод элементов указателя в алфавитном порядке в текстовый файл;
- поиск элемента предметного указателя по термину.

В демонстрационной программе предусмотреть ввод списка терминов с клавиатуры, считывание текста, для которого создается предметный указатель,

из текстового файла и запись предметного указателя в текстовый файл перед завершением программы. Количество строк на странице текста можно задать глобальной константой или ввести с клавиатуры. В текстовом файле нет переноса слов.

16. «Перевод целых чисел из одной системы счисления в другую»

Программа должна предоставлять средства для перевода целых чисел, записанных в системе счисления с основанием q ($2 \leq q \leq 16$) в систему с основанием p ($2 \leq p \leq 16$).

Операции:

- ввод числа-источника и основания его системы счисления с проверкой корректности вводимых данных;
- перевод целого числа из одной системы счисления в другую (при переводе числа из q -ой системы в 10-ую использовать схему Горнера);
- вывод числа, имеющего заданную систему счисления в виде последовательности символов, используемых в системе счисления в основания его системы счисления с проверкой корректности вводимых данных.

17. «Матричная алгебра»

Матрица должна быть представлена в памяти в виде указателя на массив указателей на строки матрицы (одномерные динамические массивы).

Операции:

- создание матрицы из чисел текстового файла (первые два числа файла – размеры матрицы);
- построчный вывод матрицы;
- разность двух матриц;
- сумма двух матриц;
- умножение матрицы на число;
- умножение матрицы на вектор;
- умножение двух матриц;
- присваивание одной матрицы другой;
- проверка равенства двух матриц;
- запись матрицы в текстовый файл (первые два числа файла – размеры матрицы).

В демонстрационной программе предусмотреть запись матриц в текстовые файлы перед завершением программы. Реализовать матрицу таким образом, чтобы можно было легко изменить тип ее элементов (использовать шаблоны функций).

18. «Средства для работы с датами»

Операции:

- ввод даты в любом формате: дд.мм.гг, дд/мм/гг, дд.мм.гггг, дд/мм/гггг;
- проверка корректности введенной даты;
- вывод даты в длинном формате (10/06/2012);
- вывод даты в американском формате (10/26/2012);
- увеличение даты на 1 день;
- увеличение даты на n дней;
- уменьшение даты на 1 день;
- уменьшение даты на n дней;
- подсчет количества дней между двумя датами;
- вычисление дня недели по дате (например, для даты 12.02.2012 день недели воскресенье).

Вычислить номер дня недели (целое число от 0 до 6, причем 0 соответствует воскресенью) можно по формуле [5]:

$$\text{Номер_дня_недели} = X \% 7$$

где $X = \text{abs}(\text{int}(2.6 * m - 0.2) + d + y/4 + y/c/4 - 2 * c)$,

m – номер месяца,

d – день месяца,

c – номер столетия,

y – номер года в столетии.

В этой формуле март имеет порядковый номер m=1, апрель m=2,..., февраль m=12. Поэтому перед использованием формулы год, месяц (и столетие) даты надо предварительно преобразовать. Например, для даты 1.2.1991 будет m=12, y=1990; для даты 31.12.1991 будет m=10, y=1991.

19. «Средства для обработки результатов опроса»

Результаты опроса n респондентов ($n \leq 3000$) по m вопросам ($m \leq 30$) с двумя альтернативными ответами записаны в текстовом файле. Ответ на вопрос «Да» в файле кодируется 1, а ответ «Нет» - 0. Обработать результаты опроса, используя для хранения данных в оперативной памяти массив из n длинных целых чисел, записывая m ответов одного респондента в m разрядов 32-х разрядного целого числа (*битовая схема хранения*). Каждый j-ответ i-го респондента хранится в виде 1 или 0 в (i-1) элементе массива в бите с номером j (0 и m+1 разряды числа для $m \leq 30$ не используются). Для выполнения операций с данными с битовой схемой хранения используются быстрые поразрядные

операции. На рис. 3.5 приведена схема хранения ответов респондентов в оперативной памяти.

Ответы на вопросы	-	1	0...	0	1	-	-	...	0	0	-	-	1	1	-...		
Значения битов	?	1	0...	0	1	?	?	...	0	0	?	?	...	1	1	?	...
Номер разряда	31	30	29...	2	1	0	31	30	...	1	0	31	30	...	1	0	...
Элемент массива	a[0]					a[1]					a[2]						
№ респондента	1					2					3						

Рисунок 3.5. Схема хранения ответов респондентов {1-респондент {1, 0, ..., 0, 1}, 2-респондент {0, ..., 0}, 3-респондент {1, ..., 1, 1}, ...}

Операции:

- чтение результатов опроса из текстового файла в двоичные разряды массива;
- подсчет ответов «да» и «нет» на j-вопрос;
- поиск вопросов, по которым большинство ответили «да»;
- поиск респондентов, которые на все вопросы ответили «нет»;
- поиск респондентов, которые на все вопросы ответили «да»;
- вывод результатов опроса на экран;
- вычисление индекса элемента массива, в котором хранится элемент со значением x (вспомогательная операция);
- вычисление номера разряда элемента массива, в котором хранится элемент со значением x (вспомогательная операция).

20. «Нахождение корня нелинейного уравнения»

Программа должна вычислять корень нелинейного уравнения методами: простых итераций, дихотомии (деления пополам), Ньютона с заданной точностью [6-7]. Программа должна предоставлять пользователю возможность выбора метода решения нелинейного уравнения и задания значения точности вычисления корня. При вычислении корня должно вычисляться количество итераций для достижения заданной точности. Результаты расчета и исходные данные (название метода вычисления и значение точности) в текущем сеансе должны накапливаться в массиве. Перед окончанием работы программы данные из этого массива должны быть добавлены в текстовый файл. Демонстрационная программа должна предоставлять пользователю возможность просмотреть результаты текущего сеанса и результаты, сохраненные в файле.

Операции:

- решение нелинейного уравнения методом итераций;
- решение нелинейного уравнения методом дихотомии;

- решение нелинейного уравнения методом Ньютона;
- вывод на экран результатов текущего сеанса;
- запись в конец текстового файла результатов текущего сеанса;
- просмотр накопленных в текстовом файле результатов.

21. «Вычисление определенного интеграла»

Программа должна вычислять приближенное значение определенного интеграла методами: средних, Симпсона, Ньютона-Котеса для $m=5$ [6-7]. Программа должна предоставлять пользователю возможность выбора метода вычисления интеграла и задания значения точности его вычисления. При вычислении интеграла должно вычисляться количество итераций для достижения заданной точности. Результаты расчета и исходные данные (название метода вычисления и значение точности) в текущем сеансе должны накапливаться в массиве. Перед окончанием работы программы данные из этого массива должны быть добавлены в текстовый файл. Демонстрационная программа должна предоставлять пользователю возможность просмотреть результаты текущего сеанса и результаты, сохраненные в файле.

Операции:

- вычисление интеграла методом средних;
- вычисление интеграла методом Симпсона;
- вычисление интеграла методом Ньютона-Котеса;
- вывод на экран результатов текущего сеанса;
- запись в конец текстового файла результатов текущего сеанса;
- просмотр накопленных в текстовом файле результатов.

4. СТРУКТУРА И СОДЕРЖАНИЕ КУРСОВОЙ РАБОТЫ

КР как письменная теоретическая работа должна иметь следующую структуру:

- титульный лист (Приложение 1);
- задание на курсовую работу (Приложение 2);
- отзыв руководителя (Приложение 3);
- содержание (оглавление);
- введение;
- основная часть (главы, разделы);
- заключение;
- список используемой литературы (источников).

КР может включать и другие разделы: «Приложения» (заполняется иллюстрациями, таблицами, диаграммами, листинг программы и т.п.); «Реферат» или «Аннотация», «Перечень сокращений». Вместе с тем элементы визуализации арифметических или аналитических выкладок должны находить свое место в тексте основных глав или заключения.

Общий объем КР не должен быть более 50 стр.

КР должна отвечать следующим требованиям:

- соответствовать установленной структуре, а по содержанию – заданию на ее выполнение;
- быть выполненной на достаточном теоретическом уровне;
- основываться на результатах самостоятельной работы;
- иметь обязательные самостоятельные выводы в заключении;
- иметь необходимый объем;
- быть оформленной в соответствии с разделом 6 настоящих методических указаний.

Титульный лист КР оформляется по установленному образцу, приведенному в Приложении 1.

Типовая форма задания на выполнение курсовой работы приведена в Приложении 2. При большом объеме пунктов 2 и 3 задания, их продолжение переносится на оборотную сторону листа задания.

В содержании (оглавлении) приводятся наименования структурных частей КР, разделов и подразделов его основной части с указанием номера страницы, с которой начинается соответствующая часть, раздел (подраздел). Пример содержания представлен в Приложении 4.

В перечне сокращений, условных обозначений, символов, единиц и терминов приводятся используемые в КР малораспространенные сокращения, условные обозначения, символы, единицы измерения и специфические термины. Если в перечне отсутствуют специфические термины или единицы измерения или условные обозначения, то данный раздел в КР не включается.

Во введении (рекомендуемый объем – 1-2 стр.) дается общая характеристика КР: обосновывается актуальность выбранной темы; определяется цель работы и задачи, подлежащие решению для её достижения; описываются объект и предмет исследования, используемые методы и информационная база исследования, а также кратко характеризуется структура КР по разделам. Пример введения КР представлен в Приложении 5.

Основная часть (рекомендуемый объем – от 10 до 40 стр.) содержит материал, необходимый для достижения цели КР и решения, поставленных задач в процессе проектирования. Содержание основной части должно соответствовать теме, указанной в задании, и полностью ее раскрывать. Обязательным для текста КР является логическая связь между разделами и последовательное развитие основной темы на протяжении всей работы, самостоятельное изложение материала, критический подход к изучаемым данным, проведение необходимого анализа, аргументированность выводов, обоснованность предложений и рекомендаций. Также обязательным является наличие в основной части КР ссылок на использованные источники.

В заключении (рекомендуемый объем – 1-2 стр.) логически последовательно излагаются теоретические выводы и/или практические предложения, которые сформулировал студент в результате выполнения КР. Пример заключения КР представлен в Приложении 9.

Список использованных источников отражает степень охвата материала при решении поставленной задачи. Приводиться не менее 5 источников, не старше 5 лет.

В приложения помещается вспомогательный материал, который при включении в основную часть работы осложняет её восприятие (таблицы вспомогательных цифровых данных, инструкции, методики, формы отчетности и других документов и т.п.). Листинг программы включается в приложение к КР.

Студент и руководитель несут ответственность за содержательную часть КР. Студент несет полную ответственность за самостоятельность выполнения и достоверность результатов КР.

5. ОСНОВНАЯ ЧАСТЬ КУРСОВОЙ РАБОТЫ

В курсовой работе должна быть разработана программа с модульной структурой: каждая подзадача должна быть оформлена в виде функции C++. Программа должна содержать удобный пользовательский интерфейс (текстовый или графический).

Основная часть курсовой работы содержит 3 раздела:

1. Техническое задание.
2. Обзор способов организации данных и обоснование выбора структуры данных для эффективного выполнения операций.
3. Описание программы.

5.1. Техническое задание

Техническое задание разрабатывается студентом совместно с руководителем курсовой работы после выбора темы. Техническое задание определяет содержание курсовой работы и требования, предъявляемые к разрабатываемой в курсовой работе программе. Техническое задание оформляется в соответствии с ГОСТ 19.201-78 «Техническое задание. Требования к содержанию и оформлению» [2]. Этот стандарт устанавливает порядок построения и оформления технического задания на разработку программы. Ниже приведен список разделов технического задания на разработку программы курсовой работы.

- 1.1. Введение.
 - 1.1.1. Наименование программы.
 - 1.1.2. Краткая характеристика области применения программы.
- 1.2. Основание для разработки.
- 1.3. Назначение разработки.
- 1.4. Требования, предъявляемые к программе.
 - 1.4.1. Требования к функциональным характеристикам программы.
 - 1.4.2. Требования к техническим средствам, используемым при работе программы.
 - 1.4.3. Требования к языкам программы и среде разработки программы.
 - 1.4.4. Требования к информационным структурам на входе и выходе программы.
- 1.5. Требования к программной документации.
- 1.6. Этапы разработки.

Пример технического задания на разработку приведен в Приложении 6.

5.2. Обзор способов организации данных и обоснование выбора структуры данных для эффективного выполнения операций

В данном разделе курсовой работы необходимо рассмотреть от 3 до 5 структур данных, подходящих для выполнения операций программы по теме курсовой работы. Оценить качественные характеристики рассмотренных структур данных и привести обоснование выбора структуры данных для эффективного выполнения операций программы.

Пример обзора способов организации данных и обоснование выбора структуры данных для эффективного выполнения операций представлен в Приложении 7.

5.3. Описание программы

Описание программы выполняется в соответствии с ГОСТ 19.402-78 «Описание программы» [3] и включает следующие разделы:

- 3.1. Общие сведения.
 - 3.1.1. Наименование программы.
 - 3.1.2. Программное обеспечение, необходимое для функционирования программы.
 - 3.1.3. Язык программирования, на котором написана программа.
- 3.2. Функциональное назначение программы (классы решаемых задач и функциональные ограничения на применение).
- 3.3. Описание логической структуры программы.
 - 3.3.1. Алгоритмы, используемые в программе.
 - 3.3.2. Структура программы с описанием функций составных частей и связей между ними.
- 3.4. Технические средства, которые используются при работе программы.
- 3.5. Вызов программы.
- 3.6. Входные данные (организация и предварительная подготовка входных данных).
- 3.7. Выходные данные.

Пример описания программы приведен в Приложении 8.

6. ТРЕБОВАНИЯ ПО ОФОРМЛЕНИЮ КУРСОВОЙ РАБОТЫ

6.1. Общие положения

Текст курсовой работы оформляется в соответствии с ГОСТ 2.105-95 Единая система конструкторской документации (ЕСКД). Общие требования к текстовым документам (с Изменением N 1, с Поправками) [4].

Все документы курсовой работы выполняются на листах белой односторонней бумаги формата А4 (210×297 мм) на одной стороне листа.

Пробел ставится после любого знака пунктуации, но не перед ним. При этом не ставятся пробелы после открывающих скобок и кавычек, так же как и перед закрывающими скобками и кавычками.

Нумерация страниц всей курсовой работы, включая приложения, сквозная. Страницы нумеруются арабскими цифрами, проставляемыми по центру внизу страницы, включая приложения.

Все параметры текста меняются во вкладках: «Формат» → «Шрифт», «Абзац», «Список». Параметры оформления текста курсовой работы (проекта) представлены в Таблице 5.1.

Текст курсовой работы печатается, соблюдая следующие размеры полей: правое — 10 мм, верхнее, нижнее — 20 мм, левое — 30 мм.

Возможно использование других начертаний (курсив, полужирный, подчеркнутый) для акцентирования внимания на определенных терминах или важной информации.

Не допускается пустое пространство на листах.

Не допускается перенос слов и сноски в конце страниц.

Таблица 5.1 — Параметры основного текста курсовой работы (проекта)

Основной текст			
Шрифт	Шрифт	<i>Шрифт</i>	Times New Roman
		<i>Начертание</i>	Обычный
		<i>Размер</i>	14 пт
		<i>Цвет текста</i>	Авто (черный)
		<i>Подчеркивание</i>	нет
		<i>Видоизменение</i>	нет
	Интервал	<i>Масштаб</i>	100%
		<i>Интервал</i>	Обычный
		<i>Смещение</i>	Нет
	Анимация	<i>Вид</i>	Нет
Абзац	Отступы и интервалы	<i>Выравнивание</i>	По ширине
		<i>Отступ слева</i>	0 см

		<i>Отступ справа</i>	0 см
		<i>Первая строка</i>	Отступ на 1,25 см.

Окончание табл. 5.1

		<i>Интервал перед</i>	0 мм
		<i>Интервал после</i>	0 мм
		<i>Междустрочный</i>	Полуторный
	Положение на странице	<i>Разбивка на страницы</i>	Запрет висящих строк

Не допускаются сокращения в тексте, исключения составляют общепринятые сокращения и сокращения, для которых в тексте была приведена полная расшифровка.

Кавычки в русском тексте (включая список использованных источников) имеют форму «...», а в английском — "...".

Различаются и правильно используются следующие знаки:

Знак дефиса «-» не разделяется пробелами и используется:

— для соединения сложных слов, содержащих дефисы (физико-математический, во-первых, кто-либо);

— для диапазонов и интервалов (100-150 см).

Знак минуса «-» используется как знак вычитания.

Знак тире «—» разделяется пробелами и используется в соответствии с правилами пунктуации (Экономика — это хозяйственная деятельность...).

При приведении цифрового материала используются только арабские цифры, за исключением общепринятой нумерации кварталов, полугодий, которые обозначаются римскими цифрами. Римские цифры и даты, обозначаемые арабскими цифрами, не должны сопровождаться падежными окончаниями. Количественные числительные в тексте пишутся также без падежных окончаний. Если в тексте необходимо привести ряд величин одной и той же размерности, то единица измерения указывается только после последнего числа. Для величин, имеющих два предела, единица измерения пишется только один раз при второй цифре.

Математические знаки, такие как «+», «-», «=», «>», «<» и так далее используются только в формулах. В тексте их следует писать словами: «плюс», «минус», «равно», «больше», «меньше».

Все иллюстрации, таблицы, листинги программ (тексты программ) располагаются в тексте непосредственно после абзаца, в котором они упоминаются впервые или на следующей странице с выравниванием по центру.

Иллюстрации, формулы (уравнения), таблицы, листинги, список использованных источников, нумеруется арабскими цифрами последовательно в преде-

лах раздела. Номер раздела и порядковый номер разделяются точкой. Пример: «Рисунок 2.7».

Ссылки в тексте на иллюстрации, таблицы, листинги, приложения пишутся в круглых скобках или включаются в предложение. При ссылке следует писать слова «Рисунок», «Таблица», «Листинг», «Приложение» с большой буквы и с указанием их порядкового номера.

Примеры: «(Таблица 5.4).» или «...представлено в Приложении А».

Курсовая работа должна быть сброшюрована.

Курсовая работа исключает содержание помарок, карандашных исправлений, пятен, трещин и загибов. Работа, представленная к защите, должна иметь все необходимые росписи на титульном листе и задании на курсовую работу.

(проект).

6.2. Построение текста

6.2.1. Разделы, подразделы, пункты

Текст курсовой работы разбивается на разделы, подразделы, пункты.

Порядковая нумерация разделов, подразделов и пунктов осуществляется арабскими цифрами, начиная с 1 в пределах всей курсовой работы, за исключением приложений. После номера раздела, подраздела и пункта ставится пробел и без знаков препинания пишется заголовок. Номер подраздела состоит из номера раздела и порядкового номера подраздела, разделенных точкой (рис. 5.1).

1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ (*раздел*)

1.1 Введение (*подраздел*)

1.1.1 Наименование программы (*пункт*)

1.1.2

...

1.2

...

2

2.1

...

Рисунок 5.1 — Пример обозначения разделов, подразделов и пунктов

Разделы « СОДЕРЖАНИЕ», « ВВЕДЕНИЕ», « ЗАКЛЮЧЕНИЕ», « СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ» не нумеруются.

6.2.2. Заголовки

Разделы, подразделы, пункты имеют заголовки, четко и кратко отражающие содержание разделов, подразделов и пунктов.

Введены следующие общие обозначения заголовков:

- заголовок раздела — Заголовок первого уровня;
- заголовок подраздела — Заголовок второго уровня;
- заголовок пунктов — Заголовок третьего и последующего уровней.

Названия заголовков разделов, подразделов и пунктов соответствуют их наименованию, указанному в содержании и в задании на курсовую работу.

Все названия заголовков печатаются с прописной буквы.

В конце названия заголовков знак препинания не ставится. Если название заголовка состоит из двух предложений, их разделяют точкой.

Параметры оформления заголовков разделов, подразделов и пунктов представлены в Таблице 5.2.

Таблица 5.2 — Параметры заголовков разделов, подразделов, пунктов

Заголовки			
Заголовок первого уровня			
Шрифт	Шрифт	Шрифт	Times New Roman
		Начертание	Полужирный
		Размер	18 пт
		Цвет текста	Авто (черный)
		Подчеркивание	нет
		Видоизменение	Все прописные
	Интервал	Масштаб	100%
		Интервал	Обычный
		Смещение	Нет
	Анимация	Вид	Нет
Абзац	Отступы и интервалы	Выравнивание	По ширине
		Отступ слева	0 см
		Отступ справа	0 см
		Первая строка	Отступ на 1,25 см
		Интервал перед	0 пт
		Интервал после	12 пт
		Междустрочный	Полуторный
	Положение на странице	Разбивка на страницы	Не отрывать от следующего С новой страницы
Заголовок второго уровня			

Шрифт	Шрифт	<i>Шрифт</i>	Times New Roman
		<i>Начертание</i>	Полужирный
		<i>Размер</i>	16 пт
		<i>Цвет текста</i>	Авто (черный)
		<i>Подчеркивание</i>	нет
		<i>Видоизменение</i>	нет
	Интервал	<i>Масштаб</i>	100%

Окончание табл. 5.2

		Интервал	Обычный
		Смещение	Нет
	Анимация	Вид	Нет
Абзац	Отступы и интервалы	Выравнивание	По ширине
		Отступ слева	0 см
		Отступ справа	0 см
		Первая строка	Отступ на 1,25 см
		Интервал перед	24 пт
		Интервал после	12 пт
		Междустрочный	Полуторный
	Положение на странице	Разбивка на страницы	Не отрывать от следующего
			Не разрывать абзац
Запрет висящих строк			
Заголовок третьего и последующих уровней			
Шрифт	Шрифт	Шрифт	Times New Roman
		Начертание	Полужирный
		Размер	14 пт
		Цвет текста	Авто (черный)
		Подчеркивание	нет
		Видоизменение	нет
	Интервал	Масштаб	100%
		Интервал	Обычный
		Смещение	Нет
Анимация	Вид	Нет	
Абзац	Отступы и интервалы	Выравнивание	По ширине
		Отступ слева	0 см
		Отступ справа	0 см
		Первая строка	Отступ на 1,25 см
		Интервал перед	24 пт
		Интервал после	12 пт
		Междустрочный	Полуторный
	Положение на странице	Разбивка на страницы	Не отрывать от следующего
			Не разрывать абзац
Запрет висящих строк			

Названия основных разделов (СОДЕРЖАНИЕ, ВВЕДЕНИЕ, ЗАКЛЮЧЕНИЕ, СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ, ПРИЛОЖЕНИЯ), курсовой работы, отформатированное стилем «Заголовок первого уровня» с выравниванием по центру (табл. 5.2), печатаются прописными буквами с выравниванием по центру.

Для форматирования документа в текстовом редакторе Word используется вкладка «Стили» (рис. 5.2). Стилем называется набор параметров форматирования, который применяется к тексту и спискам, чтобы быстро изменить их внешний вид. Стили позволяют одним действием изменить сразу всю группу атрибутов форматирования.

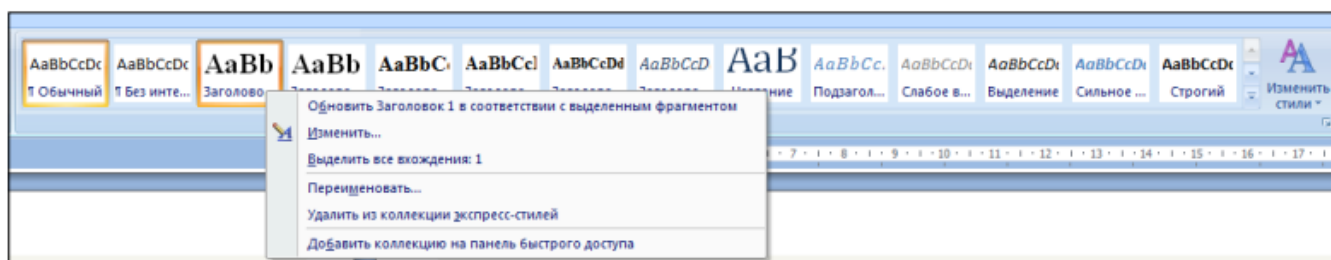


Рисунок 5.2. Использование вкладки «Стили»

Использование стилей позволяет повысить эффективность и ускорить выполнение работы. С помощью диалогового окна «Стили» можно изменять существующие стили или создавать новые стили. Пример настройки стиля заголовка первого уровня представлен на рис. 5.3.

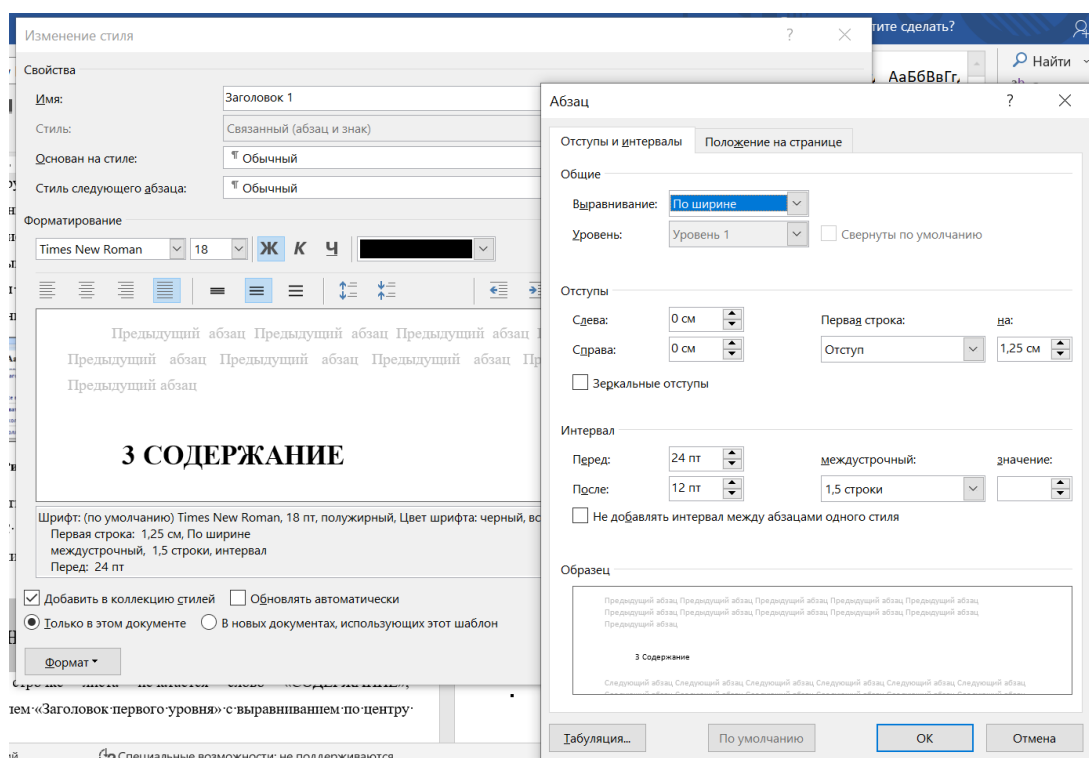


Рисунок 5.3. Стиль заголовка первого уровня

6.3. Содержание

На первой строчке листа печатается слово «СОДЕРЖАНИЕ», отформатированное стилем «Заголовок первого уровня» с выравниванием по центру (табл. 5.2).

Далее следует оглавление (в MS Word — с помощью команды главного меню «Вставка» - «Оглавление») с отображением номеров по правому краю и с использованием заполнителя «.».

Текст содержания проверяется и форматируется стилем «Основной текст курсовой работы» без отступа с выравниванием по ширине (табл. 5.1).

4 ТЕХНОЛОГИЧЕСКИЙ РАЗДЕЛ.....	74
4.1 Обоснование выбора средств разработки ИС.....	74
4.1.1 Обоснование выбора средств разработки клиентской части ИС.....	74
4.1.2 Обоснование выбора СУБД.....	75
4.2 Описание реализации клиентской части ИС	77
4.2.1 Дерево функций.....	77
4.2.2 Сценарий диалога.....	79
4.2.3 Технология работы с ИС.....	81
4.3 Описание реализации базы данных ИС	94
4.4 Рекомендации по обеспечению информационной безопасности при эксплуатации ИС	98
ЗАКЛЮЧЕНИЕ	99
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	101
ПРИЛОЖЕНИЯ	105

Рисунок 5.4. Пример оформления содержания

Названия основных разделов (ВВЕДЕНИЕ, ЗАКЛЮЧЕНИЕ, СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ, ПРИЛОЖЕНИЯ), в том числе ТЕОРЕТИЧЕСКАЯ ЧАСТЬ и ПРАКТИЧЕСКАЯ ЧАСТЬ курсовой работы, в содержании печатаются прописными буквами.

Если в содержании текст наезжает на область нахождения номеров страниц или в каком-либо пункте отсутствует заполнитель перед номером страницы, то часть строки переносится вручную на следующую строчку.

6.4. Маркированные и нумерованные списки

Маркированный, нумерованный список форматируется стилем «Основной текст курсовой работы» (табл. 5.1). Параметры оформления списков представлены в табл. 5.3.

Таблица 5.3 — Параметры маркированных и нумерованных списков

Маркированные и нумерованные списки				
Список	Маркированный, нумерованный	Знак маркера	«—», «—», «●», «■», «○», «1.», «a)»	
		Положение маркера (номера)	Отступ	1,25 см
		Положение текста	Табуляция после	2,25 см
			Отступ	2,25 см

В маркированных списках используется только один знак маркера во всем тексте курсовой работы.

Текст в маркированном списке начинается с маленькой (строчной) буквы, а заканчивается — точкой с запятой (последний пункт в списке заканчивается точкой).

Текст в нумерованном списке должен начинаться с прописной буквы и заканчиваться точкой.

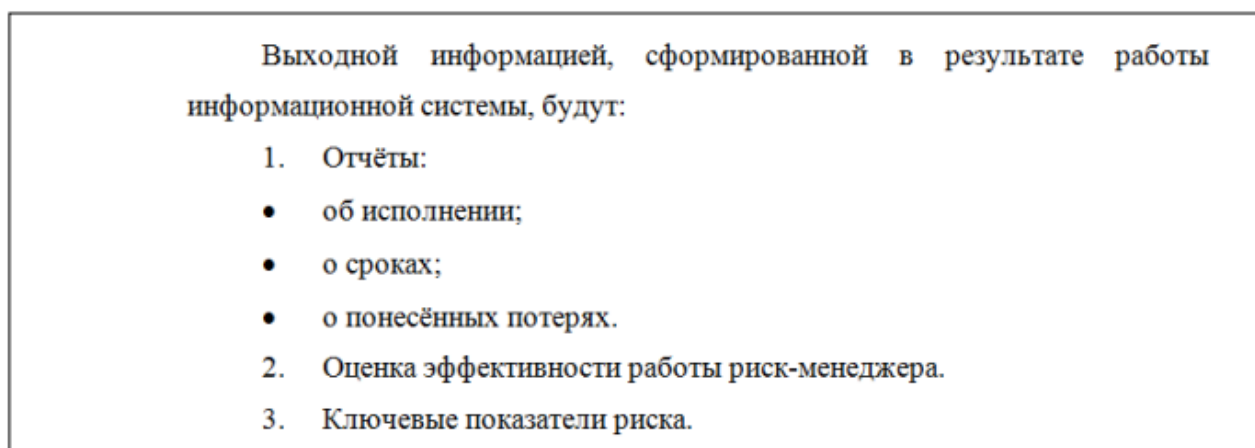


Рисунок 5.5. Пример оформления списка

6.5. Формулы, уравнения и переменные

Формулы, уравнения и переменные набираются с использованием редактора формул Microsoft Equation или Math Type. В MS Word «Вставка» → «Объект» → «Microsoft Equation»

Перед и после каждой формулы (уравнения) оставляется одна пустая строка.

Текст формулы (уравнения) располагающейся на отдельной строке, выравнивается по центру. Формат шрифтов и математический стиль устанавливаются по умолчанию. Размеры уравнений, формул и переменных изменяются в пункте «Размер» → «Определить» как показано на рис. 5.6.

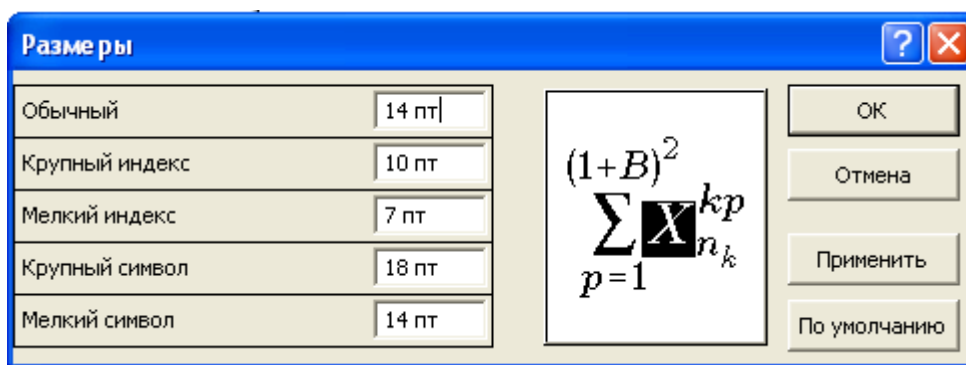


Рисунок 5.6. Размеры формул в Microsoft Equation 3.0

Если формула (уравнение) является концом предложения, ставится точка, если после формулы предложение продолжается — запятая.

Если формула (уравнение) не уместается в одну строку, оно переносится после знака равенства (=) или после знаков плюс (+), минус (-), умножения (×) или деления (:). При этом знак в начале следующей строки повторяется.

Формулы (уравнения) нумеруются последовательно в пределах раздела арабскими цифрами в круглых скобках, выровненных по правому краю. Нумеруются только те формулы, на которые имеются ссылки в тексте.

Ссылки в тексте на номер формулы даются в круглых скобках. Пример: «...в формуле (7.1)».

Пояснения значений символов и числовых коэффициентов следует приводить непосредственно под формулой в той же последовательности, в какой они даны в формуле. Первую строчку пояснения начинают без абзацного отступа со слова «где» без двоеточия после него. Значение каждого символа и числового коэффициента следует давать с новой строки, располагая символы один под другим.

Примеры:

Абсолютное снижение трудовых затрат (ΔT) :

$$\Delta T = T_0 - T_1, \quad (7.1)$$

где T_0 — трудовые затраты на обработку информации по базовому варианту;
 T_1 — трудовые затраты на обработку информации по предлагаемому варианту.

6.6. Таблицы

Параметры оформления таблиц и надписи к таблицам представлены в табл. 5.4.

Таблица 5.4 — Параметры оформления таблиц и надписей к таблицам

Таблица			
Надпись таблицы			
Шрифт	Шрифт	Шрифт	Times New Roman
		Начертание	Курсив
		Размер	12 пт
		Цвет текста	Авто (черный)
		Подчеркивание	нет
		Видоизменение	нет

Окончание табл. 5.4

	Интервал	Масштаб	100%
		Интервал	Обычный
		Смещение	Нет
	Анимация	Вид	Нет
Абзац	Отступы и интервалы	Выравнивание	По ширине
		Отступ слева	0 см
		Отступ справа	0 см
		Первая строка	нет
		Интервал перед	6 пт
		Интервал после	0 мм
		Междустрочный	Одинарный
	Положение на странице	Разбивка на страницы	Не отрывать от следующего С новой страницы
Содержание таблицы			
Шрифт	Шрифт	Шрифт	Times New Roman
		Начертание	Обычный, полужирный, курсив
		Размер	12 пт
		Цвет текста	Авто (черный)
		Подчеркивание	нет
		Видоизменение	нет
	Интервал	Масштаб	100%
		Интервал	Обычный
		Смещение	Нет
	Анимация	Вид	Нет
Абзац	Отступы и интервалы	Выравнивание	По ширине, по центру
		Отступ слева	0 см
		Отступ справа	0 см
		Первая строка	0 см
		Интервал перед	0 мм

	Интервал после	0 мм
	Междустрочный	Одинарный

Название таблицы пишется в одну строку с номером, через тире, с большой буквы, перед таблицей. Точка в конце не ставится. Пример, «Таблица 1.7 — Исходные данные».

При необходимости переноса части таблицы на другую страницу название помещается только над первой частью таблицы, нижнюю горизонтальную черту, ограничивающую таблицу, не проводят. Над другими частями пишется «Продолжение Таблицы 6.1», отформатированное стилем «Надпись. Таблица».

Автоподбор таблицы по содержанию.

Заголовки столбцов центрируются по ширине столбца, а заголовки строк выравниваются по левому краю. Заголовки граф и строк таблицы пишутся с прописной буквы в единственном числе, а подзаголовки граф — со строчной буквы, если они составляют одно предложение с заголовком, или с прописной буквы, если они имеют самостоятельное значение. В конце заголовков и подзаголовков таблиц знаки препинания не ставят. Диагональное деление ячеек таблицы не допускается.

Строка, следующая за таблицей, печатается с интервалом перед 6 пт.

Графу «№ п/п» в таблицу не включают. При необходимости нумерации показателей или других данных порядковые номера указывают в левом столбце таблицы перед их наименованием.

Таблица 2.3 — Справочник «Сотрудники финансового отдела»

Табельный номера работника	ФИО	Должность	Отдел
12225	Иванов Алексей Васильевич	Риск-менеджер	Отдел финансового контроллинга
14576	Союзов Олег Дмитриевич	Глава управления рисками	Отдел финансового контроллинга
17589	Тришин Дмитрий Олегович	Глава отдела бюджетирования	Отдел финансового контроллинга

38

Продолжение Таблицы 2.3

16475	Мартынов Андрей Фёдорович	Менеджер по бюджетированию	Отдел финансового контроллинга
15672	Столбов Павел Андреевич	Финансовый аналитик	Отдел финансового контроллинга
13468	Козлова Наталья Валерьевна	Главный бухгалтер	Бухгалтерия
16378	Кольцов Дмитрий Олегович	Менеджер по управлению денежными потоками	Отдел корпоративных финансов

Рисунок 5.7. Пример оформления переноса таблицы

6.7. Иллюстрации

Иллюстрации (графики, схемы, фотографии и т.п.) располагаются так, чтобы их было удобно рассматривать без поворота текста или путем переворачивания по часовой стрелке на 90°. В таком случае ориентация в параметрах страницы изменяется с книжной на альбомную.

Название иллюстрации пишется после номера через тире и помещается после иллюстрации. Сокращение слова «Рисунок» не допускается. В конце названия иллюстрации точка не ставится. Пример: «Рисунок 1.4 — Детали прибора».

Количество иллюстраций в пояснительной записке должно быть достаточным для пояснения излагаемого текста. Все иллюстрации и таблицы располагаются в тексте непосредственно после абзаца, в котором они упоминаются впервые или на следующей странице с выравниванием по центру. Иллюстрации должны иметь наименование.

Иллюстрации, формулы, таблицы нумеруются арабскими цифрами. Нумеровать иллюстрации необходимо последовательно в пределах раздела — номер раздела и порядковый номер разделяются точкой.

Параметры оформления надписи к иллюстрациям представлены в табл. 5.5.

Таблица 5.5 — Параметры надписи к иллюстрации

Надпись к иллюстрациям			
Шрифт	Шрифт	Шрифт	Times New Roman
		Начертание	Полужирный
		Размер	12 пт
		Цвет текста	Авто (черный)
		Подчеркивание	нет
		Видоизменение	нет
	Интервал	Масштаб	100%
		Интервал	Обычный
		Смещение	Нет
	Анимация	Вид	Нет
Абзац	Отступы и интервалы	Выравнивание	По центру
		Отступ слева	0 см
		Отступ справа	0 см
		Первая строка	нет
		Интервал перед	0 мм
		Интервал после	6 пт
		Междустрочный	Одинарный
	Положение на странице	Разбивка на страницы	Запрет висящих строк

При ссылке следует писать слово «Рисунок» с большой буквы с указанием порядкового номера. Ссылки в тексте на иллюстрации пишутся в круглых скобках или включаются в предложение, например :

- план проекта иллюстрирует диаграмма Ганта (Рисунок 3.1);
- диаграмма Ганта представлена на Рисунке 3.1

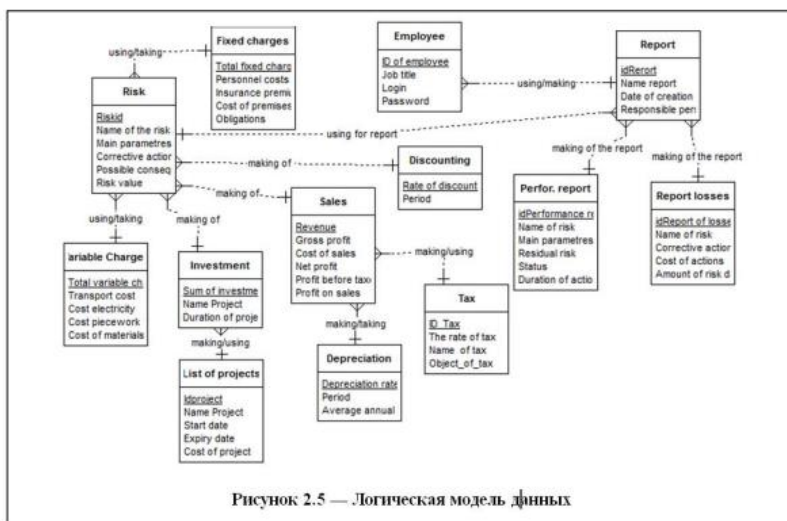


Рисунок 5.8. Пример оформления подписи иллюстрации

6.8 Текст (листинг) программ

Название листинга пишется в одну строку с номером, через тире, с большой буквы, перед листингом. Точка в конце не ставится. Пример, «Листинг 1.7 — Процедура открытия программы».

При ссылке на отдельные строки программы они нумеруются в порядке возрастания с шагом 10. После ссылки на листинг в фигурных скобках приводятся номера строк. Пример «Листинг 3.2 {30, 50, 140}», «Листинг 3.2 {20-80}».

Листинг оформляется в рамку (пример рис. 5.9).

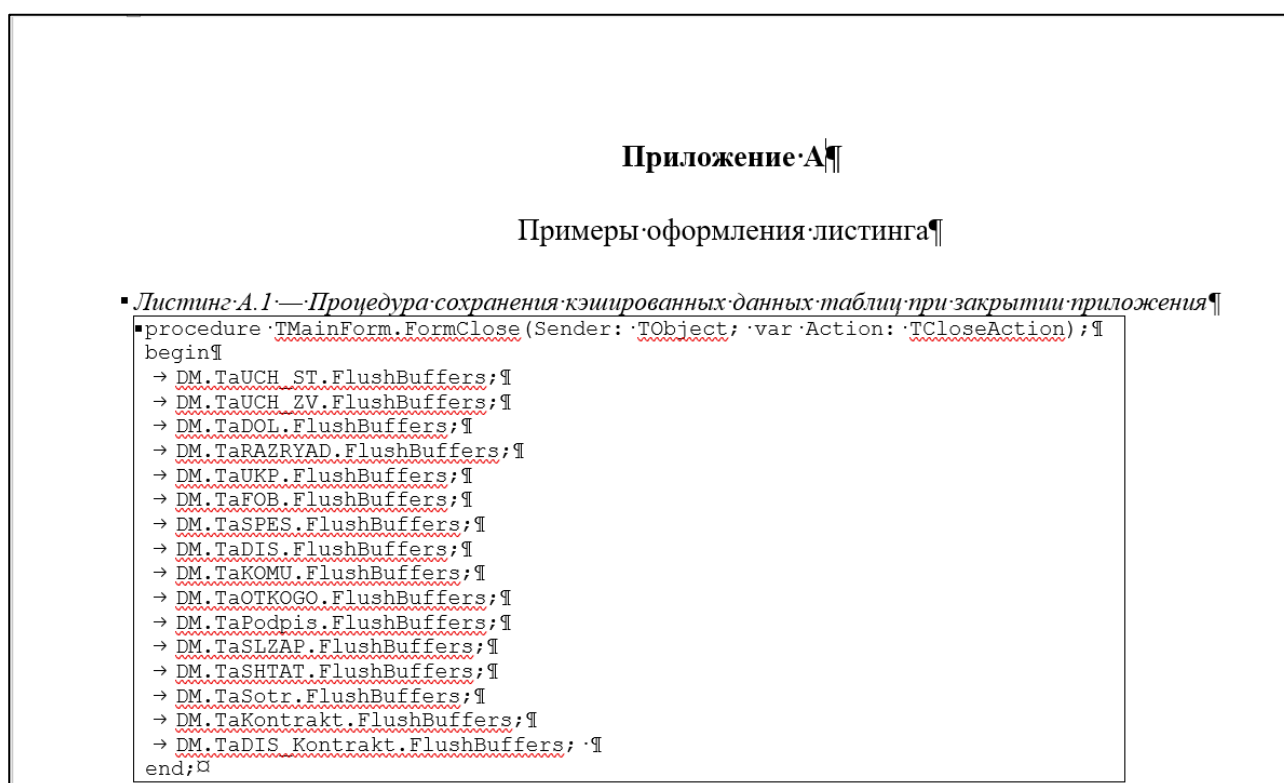


Рисунок 5.9. Пример оформления листинга программы

Параметры оформления листинга представлены в табл. 5.6.

Таблица 5.6 — Параметры листинга

Листинг			
Надпись листинга			
Шрифт	Шрифт	Шрифт	Times New Roman
		Начертание	Курсив
		Размер	12 пт
		Цвет текста	Авто (черный)
		Подчеркивание	нет
		Видоизменение	нет

	Интервал	Масштаб	100%
		Интервал	Обычный

Окончание табл. 5.6

		Смещение	Нет
	Анимация	Вид	Нет
Абзац	Отступы и интервалы	Выравнивание	По левому краю
		Отступ слева	0 см
		Отступ справа	0 см
		Первая строка	нет
		Интервал перед	6 пт
		Интервал после	0 мм
		Междустрочный	Одинарный
	Положение на странице	Разбивка на страницы	Не отрывать от следующего
			С новой страницы
Содержание листинга			
Шрифт	Шрифт	Шрифт	Courier New
		Начертание	Обычный
		Размер	10 пт
		Цвет текста	Авто (черный)
		Подчеркивание	нет
		Видоизменение	нет
	Интервал	Масштаб	100%
		Интервал	Обычный
		Смещение	Нет
	Анимация	Вид	Нет
Абзац	Отступы и интервалы	Выравнивание	По левому краю
		Отступ слева	0 см
		Отступ справа	0 см
		Первая строка	0 см
		Интервал перед	0 мм
		Интервал после	0 мм
		Междустрочный	Одинарный

6.9. Список использованных источников

На первой строчке листа печатается название « СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ», отформатированное стилем «Заголовок первого уровня» с выравниванием по центру (табл. 5.2).

Используемые источники отвечают следующим требованиям:

— общее количество используемых источников не менее 5;

— год издания источника не старше 5 лет.

В списке источники распределяются в порядке упоминания их в курсовой работе.

Перечень использованных источников оформляется стилем «Основной текст курсовой работы» (табл.5.1).

После ссылки в конце предложения ставится точка.

Ссылки в тексте на использованные источники следует давать в виде арабских цифр, заключенных в квадратные скобки, указывающих порядковый номер источника по списку, например: [1], [2]. При необходимости указываются страницы книги, статьи или другого источника, из которых взяты используемые сведения, например: [4, с.21-25].

Для некоторых городов используются следующие сокращения: Москва — «М.», Ленинград — «Л.», Санкт-Петербург — «СПб.», Киев — «К.»

Не ставятся пробелы после знаков препинания в таких случаях, как: сокращения вида «М.:», «СПб.:», используемые в списке используемых источников, а также запятая после точки в инициалах автора.

Различают правила оформления книг, журналов и Web-ссылок. Пример оформления списка используемых источников представлен на рис. 5.10.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. → Федеральный закон №127 от 23 августа 1996 г. «О науке и государственной научно-технической политике» (ред. от 23.05.2016). — URL: http://www.consultant.ru/document/cons_doc_LAW_11507/. (Дата обращения: 07.02.2019)¶
2. → Громов Т.Р. Очерки информационной технологии. — М.: ИнфоАрт. 2015. — 336 с.¶
3. → Информатика: Учебник. / Под ред. проф. Н.В. Макаровой. — М.: Финансы и статистика, 2015. — 768 с.¶
4. → Качала В.В. Предварительное обследование при реорганизации управления предприятием. // Третья Российская научно-практическая конференция «Рейнжиниринг бизнес-процессов на основе современных информационных технологий». — М.: МЭСИ, 2012. — С. 248–253.¶
5. → Надарая Э.А. Об оценке регрессии. // Теория вероятностей и ее применения. — 2010. — Т. 9. — Вып. 1. — С. 157–159.¶
6. → Пур А. Накопители XXI века. // PC Magazine. — 2013. — № 4. — С. 138–146.¶
7. → Фурсов К.С. Анализ новейших международных рекомендаций в области статистического измерения исследований и разработок (Руководство Фраскати) и возможность их адаптации в отечественной статистике. — URL: http://www.gks.ru/free_doc/new_site/rosstat/nms/prez2_1503.pdf. (Дата обращения: 07.02.2019)¶
8. → Billings S. A., Fadzil M. B., Sulley J., Johnson P. M. Identification of a non-linear difference equation model of an industrial diesel generator. // Mechanical Systems and Signal Processing. — 2015. — Vol. 2. — N. 1. — P. 59–76.¶

Рисунок 5.10. Пример оформления списка использованных источников

6.10. Приложения

Приложения оформляются как продолжение курсовой работы (проекта) на последующих ее листах.

В приложения могут выноситься:

- исходные данные для выполнения практической части курсовой работы;
- рисунки и таблицы большие по объему;
- математические выкладки и расчеты;
- методики, разработанные в процессе выполнения работы.

На первом листе раздела «Приложения» печатается полный перечень приложений с номерами и заголовками в порядке их расположения в пояснительной записке, отформатированные стилем «Основной текст курсовой работы» (табл. 5.1). Приложения обозначают заглавными буквами русского алфавита, начиная с А, за исключением букв Ё, З, Й, О, Ч, Ъ, Ы, Ь. Например: Приложение А.

Каждое приложение, при необходимости, может быть разделено на разделы. Например: Приложение А.3.

Таблицы и иллюстрации, содержащиеся в приложении, нумеруются аналогично таблицам и рисункам в основном тексте пояснительной записки. Пример: «Рисунок Б.5».

В тексте курсовой работы на все приложения даются ссылки.

Каждое приложение следует начинать с новой страницы с выравниванием по центру слова «Приложение» и его номера, отформатированного стилем «Заголовки третьего уровня» (табл. 5.2). На следующей строчке печатается заголовок приложения, выровненный по центру и отформатированный стилем «Основной текст курсовой работы» (табл. 5.1). Интервалы перед и после — 0 мм.

Листинг программы курсовой работы оформляется, как приложение.

7. ЗАЩИТА КУРСОВОЙ РАБОТЫ

Аттестация обучающихся по результатам выполнения КР должна быть проведена до начала экзаменационной сессии по расписанию. Форма промежуточной аттестации – дифференцированный зачет (зачет с оценкой), ее содержание – защита работы.

Законченные КР, подписанные обучающимся, представляются руководителю на проверку и подготовку отзыва. Срок сдачи определяется заданием на КР. Содержание проверки заключается в определении степени достижения поставленных целей, раскрытия темы КР и достоверности полученных результатов в соответствии с заданием, а также правильности оформления КР в соответствии с Рекомендациями по оформлению письменных работ обучающихся (СМКО МИРЭА 7.5.1/03.П.69), требованиями ГОСТов. Проверка КР руководителем завершается написанием (подготовкой) отзыва.

Письменный отзыв руководителя подготавливается на отдельном листе (Приложение 3).

При наличии в КР недостатков руководитель имеет право допустить ее к защите (указав на них в отзыве) или предложить обучающемуся устранить их. Обучающийся обязан доработать или переработать КР в срок, установленный руководителем с учетом сущности замечаний и объема необходимой доработки.

При наличии в КР существенных недостатков и отсутствии, по мнению руководителя, возможности ее доработки руководитель не допускает КР к защите и проставляет в экзаменационной ведомости обучающемуся неудовлетворительную оценку.

Работа, удовлетворяющая предъявляемым требованиям, с положительным отзывом руководителя, допускается к защите, о чем руководитель делает надпись на титульном листе работы.

Защита КР состоит в коротком докладе обучающегося (обычно, 5-7 минут) и в ответах на вопросы по существу КР. Вопросы могут относиться к КР, к объекту, на базе которого выполнена КР, к теории изучаемой дисциплины и т.п.

Вопросы, заданные обучающемуся во время защиты, краткая характеристика его ответов и замечания по существу работы и/или по ответам обучающегося могут быть записаны непосредственно на самой пояснительной записке.

При защите КР обучающийся должен продемонстрировать уровень сформированности компетенций, предусмотренных для закрепления данными КР в соответствии с рабочей программой дисциплины, ответить на вопросы по теме КР, а также на замечания руководителя.

При оценке КР учитывается качество устного ответа обучающегося, глубина и содержательность проработки темы, умение обосновать собственное мнение по изученным проблемам, качество анализа фактического материала, полученные выводы и рекомендации.

Оценка за КР выставляется в соответствии с показателями и критериями оценивания компетенции и используемыми шкалами оценивания, приведенными в соответствующем разделе рабочей программы дисциплины.

Обучающимся, получившим неудовлетворительную оценку за КР, предоставляется право выбора новой темы КР или, по решению руководителя, переработки прежней темы и определяется новый срок для ее выполнения.

Обучающийся, не представивший в установленный срок законченную КР или не защитивший ее, считается имеющим академическую задолженность.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. СМКО МИРЭА 7.5.1/04.И.05-18 Инструкция по организации и проведению курсового проектирования.
2. ГОСТ 19.201-78 «Техническое задание. Требования к содержанию и оформлению».
3. ГОСТ 19.402-78 «Описание программы».
4. ГОСТ 2.105-95 Единая система конструкторской документации (ЕСКД). Общие требования к текстовым документам (с Изменением N 1, с Поправками).
5. Васильев А.Н. Объектно-ориентированное программирование на C++. Издательство: Наука и Техника. Санкт-Петербург, 2016г. 543 стр.
6. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2017. — 624 с.



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий

Кафедра вычислительной техники

КУРСОВАЯ РАБОТА

по дисциплине _____

(наименование дисциплины)

Тема курсовой работы _____

Студент группы _____

(учебная группа, фамилия, имя отчество, студента)

(подпись студента)

Руководитель курсовой работы _____

(должность, звание, ученая степень)

(подпись руководителя)

Консультант (при наличии) _____

(должность, звание, ученая степень)

(подпись рецензента)

Работа представлена к защите «____» _____ 20__ г.

Допущен к защите «____» _____ 20__ г.

Москва 20__ г.



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий

Кафедра вычислительной техники

Утверждаю

Заведующий кафедрой _____

Подпись

Платонова О.В.

ФИО

«___» _____ 20__ г.

ЗАДАНИЕ

на выполнение курсовой работы по дисциплине

«_____»

Студент _____ Группа _____

Тема: _____

Исходные данные: _____

Перечень вопросов, подлежащих разработке, и обязательного графического материала: _____

Срок представления к защите курсовой работы: до «___» _____ 20__ г.

Задание на курсовую работу выдал _____ (_____)

Подпись руководителя

Ф.И.О. руководителя

Задание на курсовую работу получил «___» _____ 20__ г.

(_____)

Подпись обучающегося

Ф.И.О. исполнителя

Москва 20__ г.

ОТЗЫВ
на курсовую работу
по дисциплине «Алгоритмические основы обработки данных»

Студент(ка) _____ Группа _____

Характеристика курсовой работы

Критерий	Да	Нет	Не полностью
1. Соответствие содержания курсовой работы указанной теме			
2. Соответствие курсовой работы заданию			
3. Соответствие рекомендациям по оформлению текста, таблиц, рисунков и пр.			
4. Полнота выполнения всех пунктов задания			
5. Логичность			
6. Отсутствие фактических грубых ошибок			

Рекомендуемая оценка: удовлетворительно, хорошо, отлично

Подпись руководителя

(ФИО руководителя)

«___» _____ 20__ г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	5
1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ.....	6
1.1 Введение.....	6
1.1.1 Наименование программы	6
1.1.2 Краткая характеристика области применения программы	6
1.2 Основание для разработки	6
1.3 Назначение разработки.....	7
1.4 Требования, предъявляемые к программе.....	7
1.4.1 Требования к функциональным характеристикам программы.....	7
1.4.2 Требования к техническим средствам, используемым при работе программы.....	7
1.4.3 Требования к языкам программы и среде разработки программы.....	7
1.4.4 Требования к информационным структурам на входе и выходе программы.....	8
1.5 Требования к программной документации	8
1.6 Этапы разработки.....	8
2 ОБЗОР СПОСОБОВ ОРГАНИЗАЦИИ ДАННЫХ И ОБОСНОВАНИЕ ВЫБОРА СТРУКТУРЫ ДАННЫХ ДЛЯ ЭФФЕКТИВНОГО ВЫПОЛНЕНИЯ ОПЕРАЦИЙ	9
3 ОПИСАНИЕ ПРОГРАММЫ	12
3.1 Общие сведения.....	12
3.1.1 Наименование программы	12
3.1.2 Программное обеспечение, необходимое для функционирования программы.....	12
3.1.3 Язык программирования, на котором написана программа.....	12
3.2 Функциональное назначение программы (классы решаемых задач и функциональные ограничения на применения).....	13

3.3	Описание логической структуры программы	13
3.3.1	Алгоритмы, используемые в программе	13
3.3.2	Структура программы с описанием функций составных частей и связей между ними	15
3.4	Технические средства, которые используются при работе программы .	17
3.5	Вызов программы.....	17
3.6	Входные данные (организация и предварительная подготовка входных данных)	18
3.7	Выходные данные	18
	ЗАКЛЮЧЕНИЕ	19
	СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	20
	Приложение А.....	21

ВВЕДЕНИЕ

Любой документ должен иметь предметный указатель, чтобы были понятны его основные идеи и содержание. Когда необходимо быстро найти тему или информацию в документе, удобнее будет пользоваться предметными указателями.

Особенно в нынешнюю эпоху стремительного развития информационных технологий чтение и хранение документов в Интернете стало очень популярным. Таким образом, предметный указатель становится еще более важным для быстрого поиска определенной информации в документе. Поэтому развить программу на C ++ для создания предметных указателей с алфавитной сортировкой и поиском по заданному оглавлению. Эта программа поможет пользователям быстро найти и понять содержимое документа, особенно при работе с документами с большим объемом знаний.

Цель курсовой работы: получение практических навыков в области программирования в разработке приложения с интуитивно понятным интерфейсом по теме алгоритмов обработки данных.

Задачи, необходимые для достижения поставленной цели:

1. Рассмотреть методы и алгоритмы программирования, подходящие для разработки программы.
2. Реализовать приложение.
3. Протестировать и отладить программу.

1. ТЕХНИЧЕСКОЕ ЗАДАНИЕ

1.1 Введение

Составленное техническое задание по дисциплине «Алгоритмические основы обработки данных» является документом к курсовой работе, который отражает все этапы разработки программного продукта, а также процесс проектирования и выявление требований, предъявляемых конечному продукту.

1.1.1 Наименование программы

Название данного приложения «Работа с предметным указателем» будет напрямую связываться с темой курсовой работы: «Предметный указатель». Данное название отражает предназначение будущего приложения. Английский вариант названия: «Working with subject matter». Краткое наименование: «WWSM».

1.1.2 Краткая характеристика области применения программы

Программа предназначена для редактирования и индексации по предметному указателю. Это приложение будет полезно для всех, кто работает с большим объемом информации.

1.2 Основание для разработки

Основанием для разработки является курсовая работа по дисциплине «Алгоритмические основы обработки данных», предусмотренная учебным планом направления подготовки 09.03.01 «Информатика и вычислительная техника» профиля «Вычислительные машины, комплексы, системы и сети».

1.3 Назначение разработки

Приложение может помочь пользователям, использующим любой сервис чтения в Интернете, для быстрого поиска информации удобным способом.

1.4 Требования, предъявляемые к программе

1.4.1 Требования к функциональным характеристикам программы

В приложении должны быть реализованы следующие операции:

- создание пустого предметного указателя;
- добавление термина в предметный указатель;
- редактирование элемента предметного указателя (добавление ссылки на страницу для заданного термина);
- вывод элементов указателя в алфавитном порядке на экран;
- вывод элементов указателя в алфавитном порядке в текстовый файл;
- поиск элемента предметного указателя по термину.

1.4.2 Требования к техническим средствам, используемым при работе программы

Персональный компьютер пользователя должен быть оснащён графическим адаптером, также должна быть установлена ОС Windows (не ниже Windows 7).

1.4.3 Требования к языкам программы и среде разработки программы

Для разработки используется язык программирования C++, в качестве среды разработки выступает Visual Studio. Для разработки интерфейса пользователя задействован механизм Visual Studio.

1.4.4 Требования к информационным структурам на входе и выходе программы

В качестве входных данных программа принимает файл, содержащий предметный указатель, номер страницы указателя, список терминов для поиска.

Выходные данные представляют собой файл, содержащий алфавитный список элементов индекса или элемент индекса предметного указателя по искомому термину.

1.5 Требования к программной документации

1. Пояснительная записка оформляется в соответствии с ЛНА РТУ МИРЭА.
2. Проектная документация, составленная в соответствии с ГОСТ.

В процессе создания приложения вся проделанная работа документируется, должны быть сохранены все детали разработки, а также трудности, с которыми пришлось столкнуться. Всё вышеперечисленное должно быть отражено в пояснительной записке, которая прилагается к работе.

1.6 Этапы разработки

1. Обзор способов организации данных и обоснование выбора структуры данных для эффективного выполнения операций 02.09.2021-22.09.2021.
2. Разработка программы: 22.09.2021-30.11.2021.
3. Разработка программной документации: 01.12.2021-10.12.2021.
4. Оформление пояснительной записки: 11.12.2021-16.12.2021.
5. Защита курсовой работы: 23.12.2021-30.12.2021.

2 ОБЗОР СПОСОБОВ ОРГАНИЗАЦИИ ДАННЫХ И ОБОСНОВАНИЕ ВЫБОРА СТРУКТУРЫ ДАННЫХ ДЛЯ ЭФФЕКТИВНОГО ВЫПОЛНЕНИЯ ОПЕРАЦИЙ

Данные могут быть организованы различными способами. Тип структуры данных в программе оказывает большое влияние на ее производительность. Для того чтобы выбрать наиболее простой и эффективный способ организации данных в программе рассмотрим несколько типов структур данных.

2.1 Связный список

Связный список — одна из базовых структур данных.

Связный список состоит из группы узлов, которые вместе образуют последовательность. Также существуют двусвязные списки: в них у каждого узла есть указатель и на следующий, и на предыдущий элемент в списке.

Основные операции в связном списке включают добавление, удаление и поиск элемента в списке.

2.2 Стек

Стек — это упорядоченный набор элементов, в котором размещение новых и удаление существующих происходит с одного конца, называемого вершиной стека. Последовательность данных хранится по правилу LIFO (Last In First Out - «последним пришёл — первым ушёл»). Операция добавления элемента называется «push», удаления — «pop».

2.3 Множества

Множество — это структура данных, эквивалентная множествам в математике. Оно состоит из различных элементов заданного типа и поддерживает операции добавления элемента в множество, удаления элемента из множества,

проверка принадлежности элемента множеству. Одно и то же значение хранится в множестве только один раз.

По причине того, что множество `set` построено на списках, нельзя обратиться к определенному элементу по индексу, как в массиве или векторе. Для этого придется оперировать итераторами — структурой данных, которая используется для обращения к определенному элементу в контейнерах STL. Тип итератора должен совпадать с типом контейнера.

2.4 Массив

Массив — одна из самых простых и часто применяемых структур данных. Другие структуры данных, к примеру, стеки и очереди, являются производными от массивов. Сами массивы используются для обработки большого количества однотипных данных.

Элементы в массиве имеют свой индекс — номер элемента, по которому можно производить поиск. Существует два типа массивов: одномерные и многомерные. Первые представляют собой простейшие линейные структуры, а вторые называются вложенными и включают другие массивы. Чаще всего в программировании используются одномерные и двумерные массивы. Также существуют динамические массивы, для выделения работы с ними используются отдельные формы операторов `new` и `delete`: `new[]` и `delete[]`. Динамическое выделение массива позволяет устанавливать его длину во время выполнения программы.

2.5 Вектор

Вектор является моделью динамического массива. Другими словами, вектор — это тот же динамический массив, но который может сам управлять выделенной себе памятью. Тем самым можно создавать массивы, длина которых задается во время выполнения, без использования операторов `new` и `delete`, то есть без явного указания выделения и освобождения памяти. Использование

vector вместо ручного выделения памяти для массива позволит избежать утечек памяти, к тому же облегчит и упростит работу над кодом.

Доступ к элементам осуществляется как к обычному массиву с помощью квадратных скобок []. Все элементы вектора должны принадлежать одному типу. Для добавления нового элемента в конец вектора используется метод `push_back()`.

2.6 Выбор структуры данных

В ходе использования программы пользователь вводит в консоль текст, либо вводит название текстовых файлов, поэтому нужно воспользоваться типом данных `string` (строковый).

Чтобы реализовать хранение данных в файле, необходимо использовать файловое хранилище `fstream` для чтения, записи и хранения файла.

Для поиска или сортировки и для хранения элементов необходимо использовать динамический массив. Благодаря этому возможно использовать функции доступные в Visual Studio для сортировки, записи файлов и хранения элементов в соответствии с требованиями темы.

Благодаря преимуществу динамических массивов, можно будет вызывать элемент, хранящийся в функциях, без потери значения или удаления элемента, который нужно найти и сохранить.

3 ОПИСАНИЕ ПРОГРАММЫ

3.1 Общие сведения

В ходе выполнения курсовой работы была создана программа с консольным интуитивно понятным интерфейсом для работы с предметным указателем для операционной системы Windows. В ней выполняются все условия, обозначенные в техническом задании, и содержатся все необходимые компоненты, инструменты для корректной работы.

3.1.1 Наименование программы

Название программы: «Работа с предметным темы» или на английском языке «Working with subject matter». Оно отражает предназначение и главную функцию созданного приложения.

3.1.2 Программное обеспечение, необходимое для функционирования программы

Для корректного функционирования данного программного продукта необходимо, чтобы на персональном компьютере или ноутбуке пользователя была установлена ОС от компании Microsoft, а именно Windows (Windows 10). Также требуется наличие графического адаптера, чтобы устройство могло справляться с обработкой отображения консоли приложения. Другие требования к устройству пользователя не предусмотрены.

3.1.3 Язык программирования, на котором написана программа

Для написания программы был выбран язык программирования C++, за его доступность, понятность и высокую производительность.

3.2 Функциональное назначение программы (классы решаемых задач и функциональные ограничения на применения)

Данная программа написана для упрощения понимания предметных указателей и для удобной работы с ними. Функциональные цели приложения включают операции с текстовым файлом для работы с предметным указателем: создание предметного указателя для ввода в консоль, вывод элементов словаря в алфавитном порядке, поиск элемента указателя по термину. Также предусмотрено чтение текста для создания предметного указателя и записи предметного указателя в алфавитном порядке в другой файл, чтение предметного указателя из текстового файла для просмотра и поиска элементов.

Разработанное приложение имеет некоторые ограничения. Пользователь должен вручную ввести предметные указатели и номера страниц в панель управления, чтобы впоследствии облегчить поиск по предметному указателю. Все знаки препинания сливаются с одной буквой и не разделяются двумя пробелами. Кроме того, для пользователя будет показан неверный путь к текстовому файлу в соответствующую консоль сообщений об ошибках, и пользователю будет предложено проверить и указать соответствующий путь, по которому программа должна работать.

3.3 Описание логической структуры программы

В программе используется процедурный подход реализации алгоритмов для упрощения данного приложения. Сама программа не содержит в себе огромных методов или функций, поэтому ООП использовать нет необходимости. Исходный код программы представлен в Приложении А.

3.3.1 Алгоритмы, используемые в программе

Для написания программы необходимо подключить библиотеку «fstream» для работы с файлами, множествами соответственно, работа со строками осуществляется средствами библиотеки «string», для вызова консоли необходимо

подключить библиотеку «iostream», для манипулирования выводом программы на языке C++ — библиотеку «iomanip», для организации работы со строками, через интерфейс потоков, нужно подключить «sstream», а подключение «Windows.h» нужно для функций SetConsoleCP() и SetConsoleOutputCP() с аргументом 1251 в обеих. Библиотека <array> и <algorithm> для удобного использования динамических массивов и функций сортировки, доступных в Visual Studio.

Основными алгоритмами для работы данного приложения являются алгоритм создания частотного словаря и добавление в него элементов, алгоритм поиска элемента по слову, алгоритм записи информации в файл, алгоритм вывода информации на консоль и алгоритм запуска программы.

3.3.1.1 Алгоритм запуска программы

В начале программы пользователю предлагается текстовое меню с 6 пунктами на выбор, который осуществляется с помощью конструкции switch-case:

1. Создание пустого предметного указателя.
2. Добавление термина в предметный указатель.
3. Редактирование элемента предметного указателя (добавление ссылки на страницу для заданного термина).
4. Вывод элементов указателя в алфавитном порядке на экран.
5. Вывод элементов указателя в алфавитном порядке в текстовый файл.
6. Поиск элемента предметного указателя по термину.

В первом случае создается пустой файл, в котором могут храниться предметные указатели. Сначала создайте пустой файл с помощью библиотеки fstream, затем импортируйте предметные указатели с клавиатуры и сохраните его в файл output.

Во втором случае добавление предметного указателя в файл будет выполняться путем ввода пользователем с клавиатуры, а затем будет сохранено в файле.

В третьем случае будет реализован алгоритм записи позиций предметного указателя в начального тексте. После определения позиций будут напечатаны предметные указатели, соответствующие заданным позициям на экране.

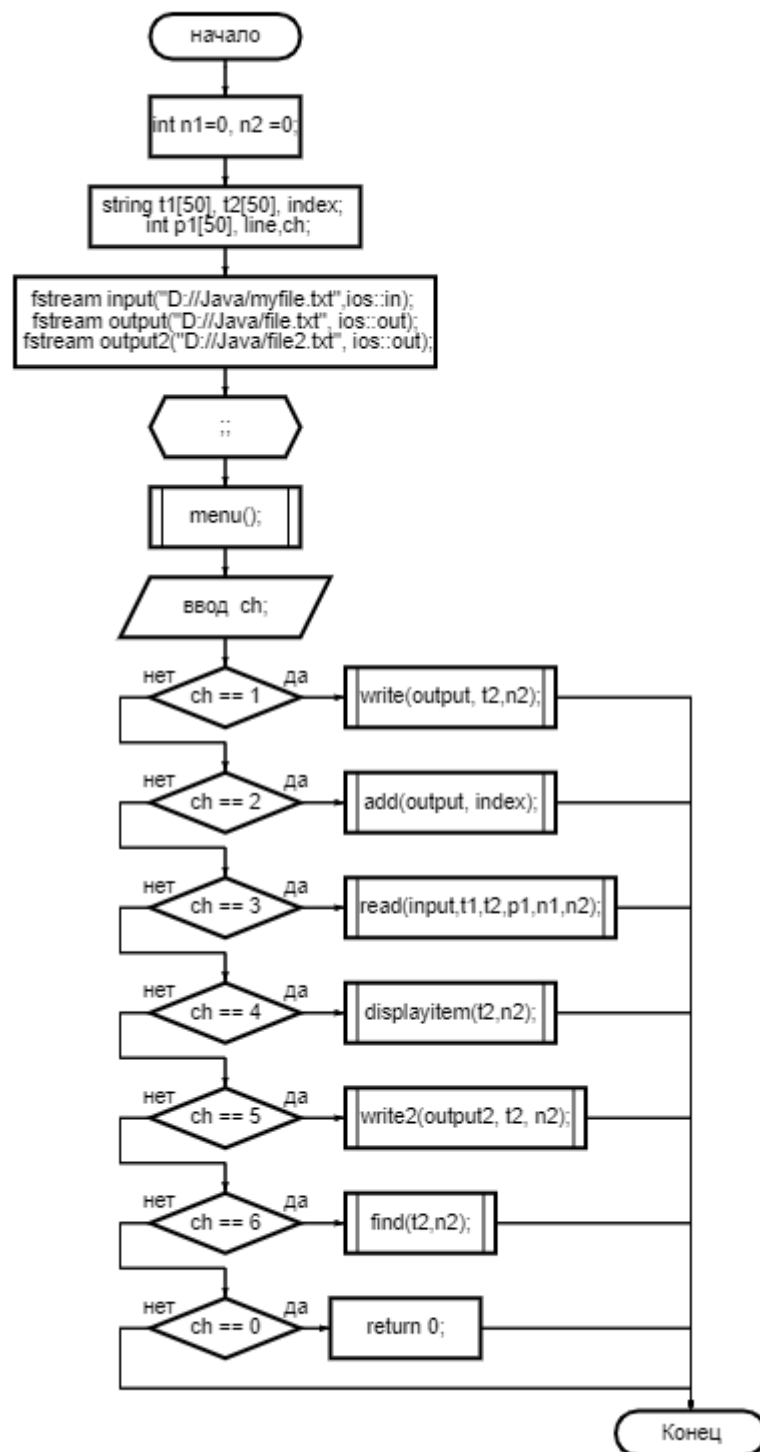


Рисунок 3.1 — Алгоритм запуска программы main()

В четвертом случае измените порядок введенных предметных указателей в алфавитном порядке. Этот алгоритм сортировки уже доступен в Visual Studio `sort ()`. После сортировки список предметных указателей будет выведен на экран.

В пятом случае после сортировки файла по алфавиту он сохраняется обратно в файл `output ()`.

И последний случай, шестой, чтобы определить, находится ли предметный указатель, который нужно найти, в файле `output ()` - файле, в котором хранятся все индексы. Будет использована функция поиска, если в файле присутствует поисковый предметный указатель, отобразится сообщение, и затем пользователь сможет продолжить поиск по предметную указателю, который ищет пользователь.

Также в основной программе необходимо предоставить текстовый файл. Количество строк на странице текста можно задать глобальной константой или вводить с клавиатуры. В текстовом файле нет переноса слов.

Блок-схема алгоритма представлена на рисунках 3.1 и 3.2.

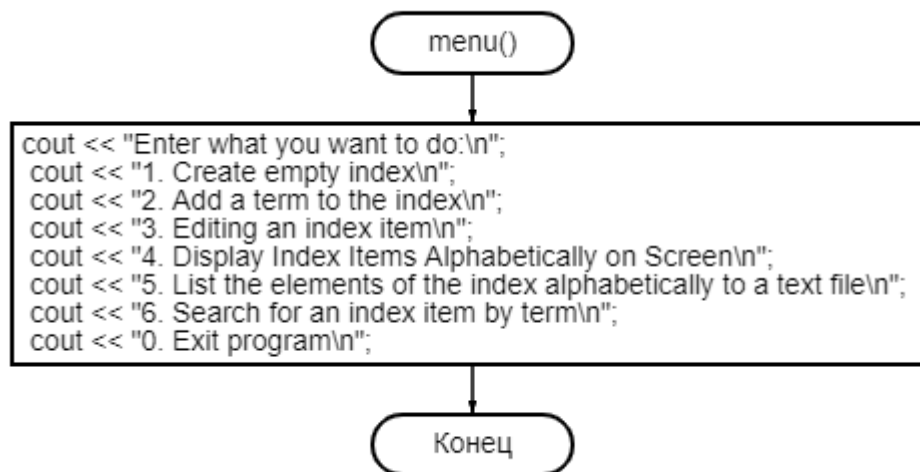


Рисунок 3.2 — Алгоритм запуска программы `menu()`

3.3.1.2 Алгоритм создание пустого предметного указателя

Этот алгоритм генерирует текст, содержащий будущие предметные указатели. Поскольку пользователю необходимо ввести предметные указатели

вручную, программа попросит ввести количество предметных указателей для сохранения. Затем используйте цикл `for`, чтобы иметь возможность вводить предметные указатели с клавиатуры. Используйте библиотеку `fstream` с `output()` файлом - сохраните весь предметный указатель, сохраните все предметные указатели, которые только что были введены с клавиатуры, в файл.

Блок-схема алгоритма пустого предметного указателя на рисунке 3.3.

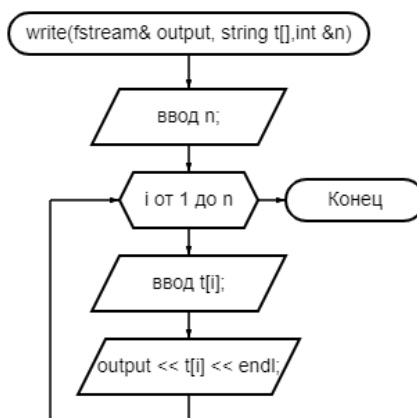


Рисунок 3.3 - Алгоритм создание пустого предметного указателя

3.3.1.3 Алгоритм добавление термина в предметный указатель

Этот алгоритм используется для добавления предметного указателя в указанный выше файл `output ()`. Сначала пользователю нужно ввести предметный указатель, который он хочет добавить, а затем предметный указатель будет сохранен в файл с помощью библиотеки `fstream`.

Блок-схема алгоритма термина в предметный указатель на рисунке 3.4.

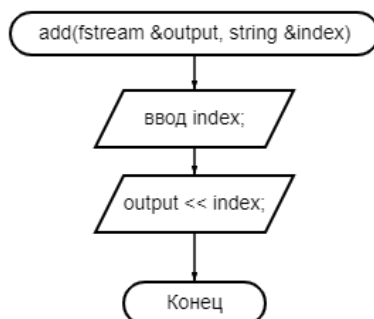


Рисунок 3.4 - Алгоритм добавление термина в предметный указатель

3.3.1.4 Алгоритм редактирование элемента предметного указателя (добавление ссылки на страницу для заданного термина)

Алгоритм редактирования элемента указателя темы (добавление ссылки на страницу по заданному термину) будет настроен следующим образом. Сначала программа будет считывать исходный текстовый файл ввода, чтобы найти номер страницы предметного указателя в файле output().

Программа использует библиотеку fstream, чтобы открыть файл и проверить: если пользователь вводит неправильный адрес ссылки текстового файла, он получит уведомление и попросит повторно ввести файл. В противном случае, если адрес файла ссылки правильный, программа начнет работать. Программа сначала использует цикл for и два динамических массива, динамический массив (t1 []), в котором хранятся все предметные индексы, содержащиеся в файле output (), который вызывается в программе для использования. Второй динамический массив (p []) будет хранить все страницы, на которых появился каждый предметный указатель. Затем массив for будет перебирать все буквы исходного текста, когда он находит предметный индекс, хранящийся в динамическом массиве (t1 []) на любой странице, он немедленно сохраняется в динамическом массиве p [].

После завершения цикла for для определения положения всех предметных указателей программа добавит еще два цикла for для отображения постраничной позиции каждого предметного указателя в тексте на экране.

Блок-схема алгоритма элемента предметного указателя (добавление ссылки на страницу для заданного термина) на рисунке 3.5.

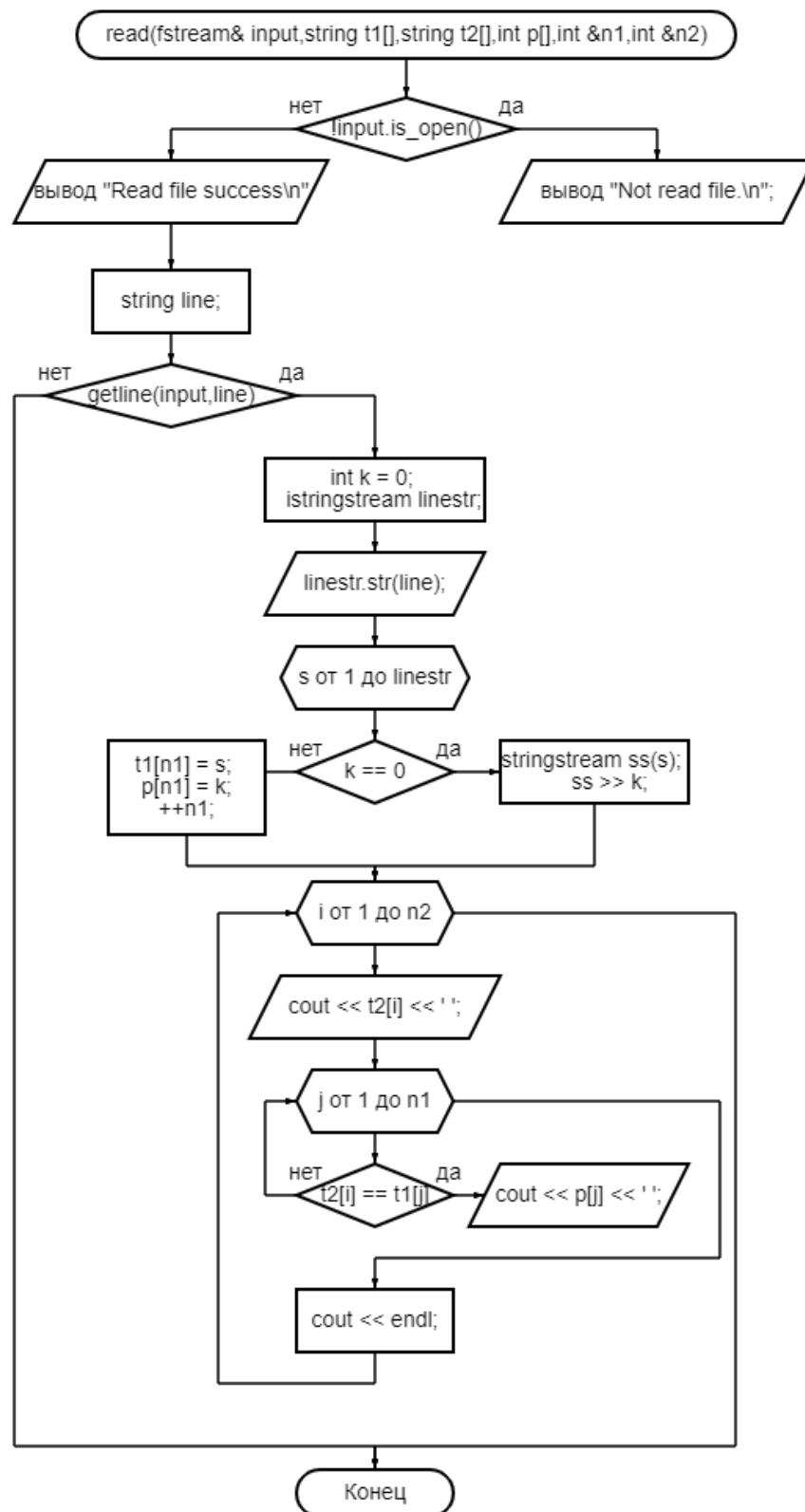


Рисунок 3.5 - Алгоритм редактирование элемента предметного указателя (добавление ссылки на страницу для заданного термина)

3.3.1.5 Алгоритм вывод элементов указателя в алфавитном порядке на экран

Алгоритм отображения записей указателя в алфавитном порядке на экране будет запрограммирован следующим образом. В Visual Studio вызовите библиотеку <algorithm>, чтобы использовать алгоритм сортировки на лету. Также используйте библиотеку <array> для хранения динамических массивов всех предметных указателей, которые необходимо отсортировать. После того, как сортировка будет выполнена в алгоритме `sort()`, используйте цикл `for` для получения результатов.

Блок-схема алгоритма элементов указателя в алфавитном порядке на экран на рисунке 3.6.

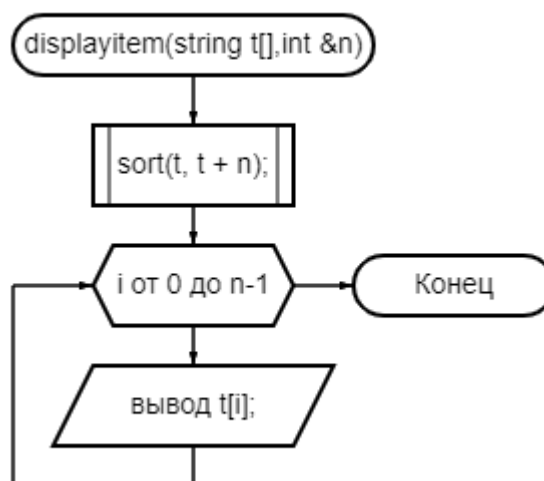


Рисунок 3.6 - Алгоритм вывод элементов указателя в алфавитном порядке на экран

3.3.1.6 Алгоритм вывод элементов указателя в алфавитном порядке в текстовый файл

В алгоритме, который выводит элементы индекса в алфавитном порядке в текстовый файл, программа просто повторно использует библиотеку `fstream` и цикл `for` для сохранения всех отсортированных предметных индексов в файл `output()`.

Блок-схема алгоритма элементов указателя в алфавитном порядке в текстовый файл на рисунке 3.7

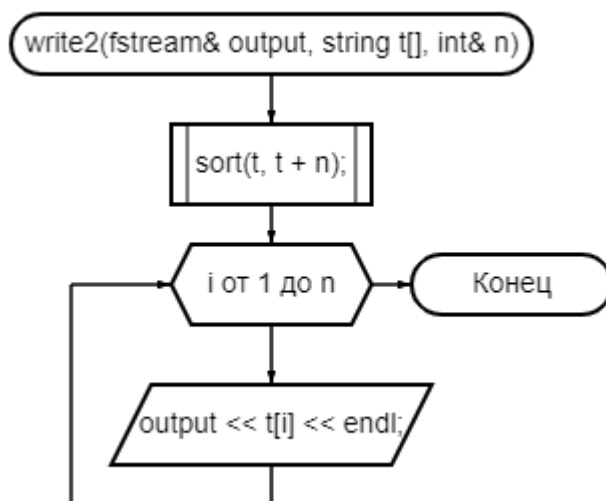


Рисунок 3.7 - Алгоритм вывод элементов указателя в алфавитном порядке в текстовый файл

3.3.1.7 Алгоритм поиск элемента предметного указателя по термину

Алгоритм поиска элемента предмета по термину. Используйте функцию `bool`, чтобы определить, присутствует ли предметный указатель, который нужно искать. Вам нужно использовать динамический массив `t []` и цикл `for` для перебора элементов предметного индекса. Если в файле есть предметный указатель, который нужно искать, на экране отобразится сообщение о том, что предметный указатель, который нужно искать, и наоборот.

Блок-схема алгоритма поиска элемента предметного указателя по термину. на рисунке 3.8.

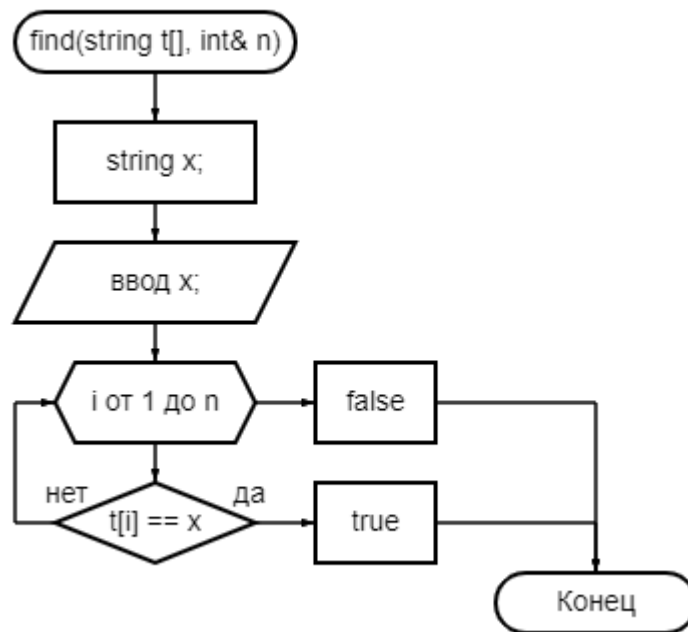


Рисунок 3.8 - Алгоритм поиск элемента предметного указателя по термину.

3.3.2 Структура программы с описанием функций составных частей и связей между ними

Для удобного взаимодействия с приложением было создано несколько функций, которые вызываются при разных выбора пунктов меню в главной функции запуска программы `int main()`, в зависимости от того, происходит работа частотного словаря с введенным текстом в консоли или же с текстовым файлом:

- `write(output, t2, n2)` - функция создание предметного указателя пуст и записывает предметный указатель, введенные с клавиатуры
- `add(output, index)` — функция добавление термина в предметный указатель;
- `read(input, t1, t2, p1, n1, n2);` — функция редактирование элемента предметного указателя;
- `displayitem(t2, n2)` — функция вывод элементов указателя в алфавитном порядке на экран.
- `write2(output2, t2, n2)` — функция вывод элементов указателя в алфавитном порядке в текстовый файл.

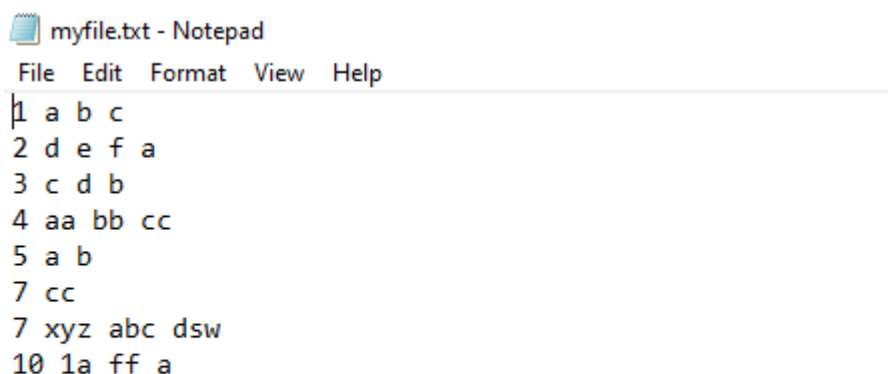
- `find(t2, n2)` — функция поиск элемента предметного указателя по термину.

3.4 Технические средства, которые используются при работе программы

Для запуска программы пользователю необходимо открыть предоставляемый пользователю файл «.exe». Как было отмечено ранее, для корректной работы продукта необходимо наличие графического адаптера и операционной системы Windows.

3.5 Вызов программы

Исходный текстовый файл (Рисунок 3.9):



```
myfile.txt - Notepad
File Edit Format View Help
1 a b c
2 d e f a
3 c d b
4 aa bb cc
5 a b
6 c c
7 xyz abc dsw
8
9
10 1a ff a
```

Рисунок 3.9 — Исходный текстовый файл

При запуске программного продукта появляется главное окно приложения (Рисунок 3.10), содержащие текстовое меню со всеми указанными в техническом задании операциями над частотным словарем. При выборе определенного пункта меню производятся необходимые операции.

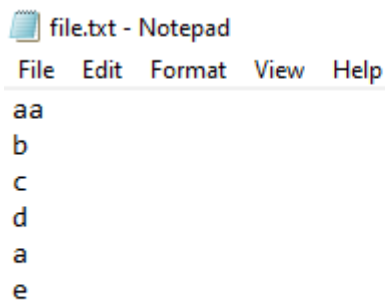
```
Enter what you want to do:
1. Create empty index
2. Add a term to the index
3. Editing an index item
4. Display Index Items Alphabetically on Screen
5. List the elements of the index alphabetically to a text file
6. Search for an index item by term
0. Exit program
```

Рисунок 3.10 — Главное окно приложения

Далее будет показана работа всех выбранных пунктов меню на рисунках 3.11, 3.13, 3.15, 3.16, 3.18, а на рисунках 3.12, 3.14, 3.17 представлены примеры текстовых файлов, куда записывалась или откуда считывалась информация предметного указателя.

```
1
Enter number index master: 6
Enter Subject index: aa
Enter Subject index: b
Enter Subject index: c
Enter Subject index: d
Enter Subject index: a
Enter Subject index: e
```

Рисунок 3.11 – Создание и вывод предметного указателя с клавиатуры



```
file.txt - Notepad
File Edit Format View Help
aa
b
c
d
a
e
```

Рисунок 3.12 - Пример работы file.txt

```
2
Add Subject index : ff
```

Рисунок 3.13 – Добавление элемента в предметный указатель с клавиатуры

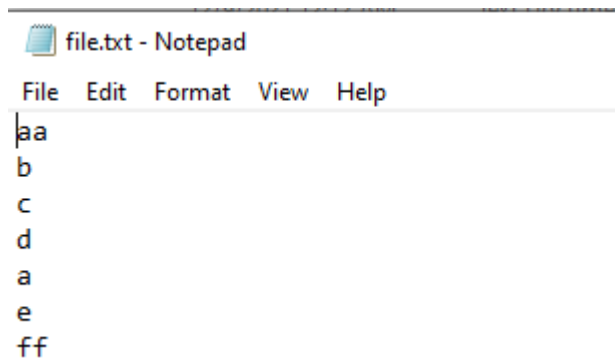


Рисунок 3.14 – Сохранение предметного указателя в текстовый файл file.txt

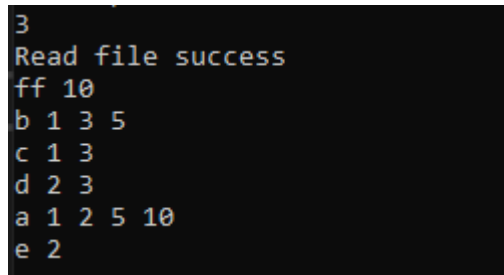


Рисунок 3.15 - Редактирование элементов предметного указателя

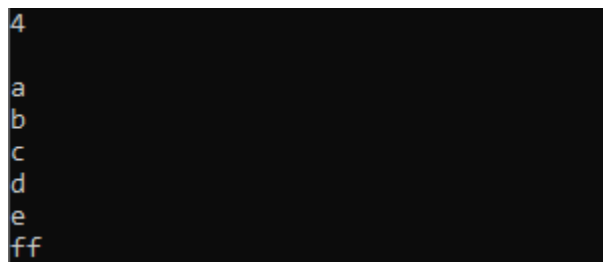


Рисунок 3.16 – Сортировка и вывод элементов указателя в алфавитном порядке на экран

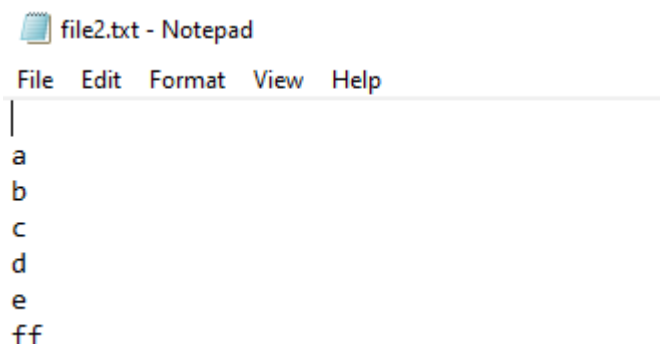


Рисунок 3.17 - Сохранение изменений предметного указателя в текстовый файл file.txt

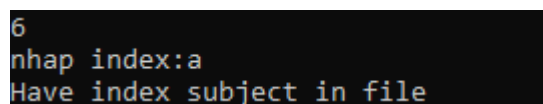


Рисунок 3.18 - Поиск элемента предметного указателя по термину

3.6 Входные данные (организация и предварительная подготовка входных данных)

В качестве входных данных программа принимает текст, который вводит пользователь. Также программа может принять данные из текстовых файлов, расположение которых указывает пользователь в консоли.

3.7 Выходные данные

Выходные данные - это номер строки, который включает предметный указатель и номер страницы предметный указатель.

ЗАКЛЮЧЕНИЕ

На протяжении всего процесса проектирования и создания программного продукта были получены практические навыки в области структур данных, которые используются для хранения и обработки различных типов данных. Также были освоены алгоритмы работы с файлами в C++.

Успешно выполнены поставленные задачи: создание программы, позволяющей осуществлять операции над частотным словарём, отладка и проверка работоспособности конечной программы, применение алгоритмов для работы с частотным словарем. Среди упомянутых операций: составление частотного словаря по введённому тексту, вывод элементов в алфавитном порядке и поиск элемента частотного словаря по слову, сохранение данных в файл и считывание их из текстового файла.

Сведения об авторах

Платонова Ольга Владимировна, к.т.н, доцент, заведующая кафедрой Вычислительной техники Института информационных технологий РТУ МИРЭА.

Асадова Юлия Сергеевна, старший преподаватель кафедры Вычислительной техники Института информационных технологий РТУ МИРЭА.

Расулов Мирзо Максудович, старший преподаватель кафедры Вычислительной техники Института информационных технологий РТУ МИРЭА.