

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«Уфимский университет науки и технологий»

Кафедра технической кибернетики

100	7	8	9	10	11	12	13	14	15	16	17
90											
80											
70											
60											
50											
40											
30											
20											
10											
0											

Проектирование классов с использованием
механизмов наследования, перегрузки
операций ввода/вывода и присваивания в
языке С++ для обработки файлов данных

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине «Языки программирования»

3271.202491.000 ПЗ

(обозначение документа)

Группа	ИВТ-ИВСС-201Б	Фамилия И. О.	Подпись	Дата	Оценка
Студент		Черников А. Ю.			
Консультант		Ракипова А. С.			
Принял		Ракипова А. С.			

Уфа 2025

ЗАДАНИЕ

на курсовой проект по дисциплине «Операционные системы»

Студент Черников А. Ю. Группа ИВТ-ИВСС-201Б Консультант Рыжов Г. И.

1. Тема курсового проекта

Проектирование гипотетической операционной системы с заданными характеристиками

2. Основное содержание: на основе заданного индивидуального задания спроектировать гипотетическую компьютерную операционную систему.

Индивидуальное задание:

Режим работы компьютера: однопрограммный пакетный.

Архитектура операционной системы: монолитное ядро с отдельной *User Interface* оболочкой.

Подсистема управления процессами:

- алгоритм планирования процессов: невытесняющий *SPN (Shortest Process Next)*;
- алгоритм организации взаимодействия процессов: алгоритм Петерсона;
- задача синхронизации процессов: задача «производителей и потребителей»;
- способ борьбы с тупиками: обход тупиков.

Подсистема управления памятью: распределение памяти динамическими разделами (без использования внешней памяти).

Подсистема управления файлами:

- иерархия каталогов: сеть;
- логическая организация файловой системы: последовательность логических записей переменной длины;
- физическая организация файловой системы: связанный список блоков.

Загрузка операционной системы: загрузка с флеш-диска *USB*.

Запуск прикладных программ: запуск с жесткого диска компьютера.

Структура управляющих блоков базы данных операционной системы:

- формирование базы данных операционной системы;
- схема взаимодействия управляющих блоков базы данных операционной системы.

3. Требования к оформлению. Пояснительная записка должна быть оформлена в редакторе *Microsoft Word* в соответствии с требованиями ГОСТ.

В пояснительной записке должны содержаться следующие разделы:

Титульный лист.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	4
1 АННОТАЦИЯ	6
2 ОСНОВНАЯ ЧАСТЬ	7
2.1 Описание разработанных функций	7
3 ЛИСТИНГ ПРОГРАММЫ.....	13
3.1 Главный модуль программы main.cpp.....	13
3.2 Модуль структур. structs.h.....	16
3.3 Модуль глобальных функций. functions.h и .cpp	17
3.4 Модуль класса classPharmacy_A. class_A.h и .cpp	18
3.5 Модуль класса classPharmacy_B. class_B.h и .cpp.....	27
3.6 Модуль класса classPharmacy_C. class_C.h и .cpp.....	38
4 ТЕСТИРОВАНИЕ ПРОГРАММЫ.....	48
4.1 Работа с исходным массивом данных	49
4.2 Работа с перечнями данных.....	56
4.3 Работа с перечнями данных по поиску определенного значения	62
ЗАКЛЮЧЕНИЕ	66
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ	67

					<h1>3271.202491.000 ПЗ</h1>		
Изм.	Лист	№ докум.	Подп.	Дата			
Разраб.		Черников А.Ю.			Проектирование классов с использованием механизмов наследования, перегрузки операций ввода/вывода...		
Пров.		Ракипова А.С					
					ФГБОУ ВО «УУНУТ» ИВТ-ИВСС-201Б		
Н.контр.							
Утв.							
					Лит.	Лист	Листов
					у	3	67

ВВЕДЕНИЕ

В современном программировании объектно-ориентированный подход является ключевым методом организации кода, обеспечивая гибкость, повторное использование и удобство сопровождения программ.

В рамках курсового проекта будет разработана программа на языке C++, использующая основные особенности объектно-ориентированного программирования, а именно использование классов как основного механизма обработки данных.

Целью создания курсового проекта является приобретение практических навыков проектирования классов в программах на языке программирования C++. Проект предназначен для демонстрации работы классов, использующих механизмы наследования, перегрузки операций.

В курсовом проекте должны быть решены следующие задачи:

1. Провести анализ заданных требований к многомодульной программе.
2. Спроектировать структурный тип PHARMACY, содержащий поля: номер аптеки, дата открытия, фамилия и инициалы владельца.

3. Спроектировать класс, предназначенный для работы с исходным массивом данных, содержащий поле адрес динамического массива структур и поле фактическое количество элементов в этом массиве.

4. Спроектировать класс, предназначенный для работы с перечнем, содержащий поле адрес динамического массива структур (перечня) и поле фактическое количество элементов в перечне.

5. Спроектировать класс, предназначенный для работы с массивом данных, содержащий поле адрес динамического массива структур (результата поиска) и поле фактическое количество элементов в массиве.

6. Для каждого класса определить следующие специальные методы:

- конструктор без параметров (по умолчанию);
- конструктор копирования;
- деструктор;

- перегруженная операция присваивания;
- перегруженная операция вывода на экран объекта класса.

7. Для каждого класса определить дружественные функции.

8. Для каждого класса определить методу обработки данных.

9. Спроектировать основную программу в виде меню, демонстрирующее работу всех методов класса.

					3271.202491.000 ПЗ	Лист
						5
Изм.	Лист	№ докум.	Подп.	Дата		

1 АННОТАЦИЯ

Разработали многомодульную программу, предназначенную для работы с исходным массивом данных путем использования классов как основополагающего инструмента.

Описали все разработанные функции в табличном формате, продемонстрировали код и работоспособность этих функций.

Полностью протестировали программу, проверив на корректную работу все пункты меню программы.

					3271.202491.000 ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подп.	Дата		

2 ОСНОВНАЯ ЧАСТЬ

2.1 Описание разработанных функций

2.1.1 Функции класса classPharmacy_A

Таблица 1. Перечень функций класса classPharmacy_A

Название функции	Параметры функции	Назначение функции
Специальные методы класса		
classPharmacy_A	-	Конструктор класса: инициализация экземпляра класса
classPharmacy_A	classPharmacy_A &objectPharmacy	Конструктор копирования одного экземпляра класса в другой
~classPharmacy_A	-	Деструктор: удаление массива данных экземпляра класса
Методы класса		
inputFile	-	Ввод данных из файла в массив экземпляра класса
outputData	-	Вывод массива в терминал
outputFile	-	Запись данных массива в текстовый файл
addPharmacy	-	Добавление записи в массив
deletePharmacy	-	Удаление записи из массива
sortNumber	-	Сортировка массива по номеру аптеки
sortName	-	Сортировка массива по ФИО владельца
sortDate	-	Сортировка массива по дате открытия
exitProgram	-	Выход из программы

Продолжение таблицы 1

Операторы		
operator =	classPharmacy_A &objectPharmacy	Перегруженная операция присваивания: аналог конструктора копирования
Дружественные функции		
operator <<	ostream &out, classPharmacy_A &objectPharmacy	Перегруженная операция вывода данных в терминал, аналог метода inputData()

2.1.1 Функции класса classPharmacy_B

Таблица 2. Перечень функций класса classPharmacy_B

Название функции	Параметры функции	Назначение функции
Специальные методы класса		
classPharmacy_B	-	Конструктор класса: инициализация экземпляра класса
classPharmacy_B	classPharmacy_B &objectPharmacy	Конструктор копирования одного экземпляра класса в другой
~classPharmacy_B	-	Деструктор: удаление массива данных экземпляра класса
Методы класса		
outputData	string &id	Вывод массива в терминал
outputFile	string &id	Запись данных массива в текстовый файл
sortNumber	-	Сортировка перечня по номеру аптеки
sortCount	-	Сортировка перечня количеству записей
sortDate	-	Сортировка перечня по дате открытия
makePerech	string &id	Формирование перечня

Продолжение таблицы 2

Операторы		
operator =	classPharmacy_B &objectPharmacy	Перегруженная операция присваивания: аналог конструктора копирования
Дружественные функции		
operator <<	string &id, classPharmacy_B &objectPharmacy	Перегруженная операция вывода данных в терминал, аналог метода inputData()

2.1.2 Функции класса classPharmacy_C

Таблица 3. Перечень функций класса classPharmacy_C

Название функции	Параметры функции	Назначение функции
Специальные методы класса		
classPharmacy_C	-	Конструктор класса: инициализация экземпляра класса
classPharmacy_C	classPharmacy_C &objectPharmacy	Конструктор копирования одного экземпляра класса в другой
~classPharmacy_C	-	Деструктор: удаление массива данных экземпляра класса
Методы класса		
outputData	string &id	Вывод массива в терминал
outputFile	string &id	Запись данных массива в текстовый файл
sortNumber	-	Сортировка перечня по номеру аптеки
sortName	-	Сортировка перечня по ФИО владельцев
makeSearch	string &id	Формирование перечня по поиску
Операторы		
operator =	classPharmacy_C &objectPharmacy	Перегруженная операция присваивания: аналог конструктора копирования
Дружественные функции		
operator <<	string &id, classPharmacy_C &objectPharmacy	Перегруженная операция вывода данных в терминал, аналог метода inputData()

2.1.3 Глобальные функции

datecomp	structDate date_1, structDate date_2	Операция сравнения дат
validateInput()	-	Проверка входных данных на соответствие типу данных

3 ЛИСТИНГ ПРОГРАММЫ

3.1 Главный модуль программы main.cpp

```
1  #include <iostream>
2  #include <string>
3  #include <conio.h>
4
5  #include "structs.h"
6  #include "../classes/class_A.h"
7  #include "../classes/class_B.h"
8  #include "../classes/class_C.h"
9
10 using namespace std;
11
```

Рисунок 1. Набор подключаемых библиотек и модулей

Подключаем стандартные библиотеки:

- `iostream` — для ввода/вывода;
- `string` — для работы со строками;
- `conio.h` — предоставляет функции для работы с клавиатурой (например, `getch()`).

Подключаем пользовательские заголовочные файлы модулей:

- `structs.h` — содержит определения структур данных;
- `class_A.h`, `class_B.h`, `class_C.h` — заголовки, содержащие определения классов А, В и С соответственно.

Используем пространство имен `std` для обращения к сущностям без явного указания пространства имен.

```

13  int main() {
14      classPharmacy_C pharmacyData;
15      int choice;
16
17      while (true) {
18          cout << "\n";
19          cout << "1. Ввод исходного массива из файла;\n";
20          cout << "2. Вывод массива в терминал;\n";
21          cout << "3. Вывод массива в текстовый файл;\n";
22          cout << "4. Добавление записи;\n";
23          cout << "5. Удаление записи;\n";
24          cout << "6. Сортировка массива по номеру аптеки;\n";
25          cout << "7. Сортировка массива по ФИО владельца;\n";
26          cout << "8. Сортировка массива по дате открытия;\n";
27          cout << "-----\n";
28          cout << "9. Формирование перечня номеров аптек;\n";
29          cout << "10. Вывод перечня дат открытия в терминал;\n";
30          cout << "11. Вывод перечня дат открытия в текстовый файл;\n";
31          cout << "-----\n";
32          cout << "12. Формирование перечня дат открытия;\n";
33          cout << "13. Вывод перечня дат открытия в терминал;\n";
34          cout << "14. Вывод перечня дат открытия в текстовый файл;\n";
35          cout << "-----\n";
36          cout << "15. Формирование перечня ФИО владельцев;\n";
37          cout << "16. Вывод перечня ФИО владельцев в терминал;\n";
38          cout << "17. Вывод перечня ФИО владельцев в текстовый файл;\n";
39          cout << "-----\n";
40          cout << "18. Формирование перечня по поиску ФИО владельца;\n";
41          cout << "19. Вывод перечня по поиску ФИО владельца в терминал;\n";
42          cout << "20. Вывод перечня по поиску ФИО владельца в текстовый файл;\n";
43          cout << "-----\n";
44          cout << "21. Формирование перечня по поиску даты открытия;\n";
45          cout << "22. Вывод перечня по поиску даты открытия в терминал;\n";
46          cout << "23. Вывод перечня по поиску даты открытия в текстовый файл;\n";
47          cout << "-----\n";
48          cout << "24. Выход из программы;\n";
49          cout << "Ваш выбор (1-24): ";
50          cin >> choice;
51
52          if (cin.fail()) {
53              string s;
54              cin.clear();
55              cin >> s;
56              cout << "Это не пункт меню.\n";
57              getch();
58              continue;
59          }
60      }

```

Рисунок 2. Реализация функции main. Вывод меню функций на экран

Объявляем экземпляр класса pharmacyData класса classPharmacy_C, который наследует сущности классов classPharmacy_A и classPharmacy_B.

Выводим список возможностей программы для обработки данных и запрашиваем номер меню для вызова соответствующей функции. Проверяем введенное значение на соответствие числовому типу данных.

```

61     string id;
62     switch (choice) {
63         case 1: pharmacyData.inputFile(); break;
64         case 2: cout << (classPharmacy_A&) pharmacyData; break;
65         case 3: pharmacyData.classPharmacy_A::outputFile(); break;
66         case 4: pharmacyData.addPharmacy(); break;
67         case 5: pharmacyData.deletePharmacy(); break;
68         case 6: pharmacyData.classPharmacy_A::sortNumber(); break;
69         case 7: pharmacyData.classPharmacy_A::sortName(); break;
70         case 8: pharmacyData.classPharmacy_A::sortDate(); break;
71
72         case 9: pharmacyData.makePerech(id="number"); break;
73         case 10: {id = "number"; id << (classPharmacy_B&)pharmacyData;} break;
74         case 11: pharmacyData.classPharmacy_B::outputFile(id="number"); break;
75
76         case 12: pharmacyData.makePerech(id="date"); break;
77         case 13: {id = "date"; id << (classPharmacy_B&)pharmacyData;} break;
78         case 14: pharmacyData.classPharmacy_B::outputFile(id="date"); break;
79
80         case 15: pharmacyData.makePerech(id="name"); break;
81         case 16: {id = "name"; id << (classPharmacy_B&)pharmacyData;} break;
82         case 17: pharmacyData.classPharmacy_B::outputFile(id="name"); break;
83
84         case 18: pharmacyData.makeSearch(id="name"); break;
85         case 19: {id = "name"; id << pharmacyData;} break;
86         case 20: pharmacyData.classPharmacy_C::outputFile(id="name"); break;
87
88         case 21: pharmacyData.makeSearch(id="date"); break;
89         case 22: {id = "date"; id << pharmacyData;} break;
90         case 23: pharmacyData.classPharmacy_C::outputFile(id="date"); break;
91
92         case 24: pharmacyData.exitProgram(); break;
93         default: cout << "Нет такого пункта в меню.\n"; getch();
94     }
95 }
96
97 return 0;
98 }
99

```

Рисунок 3. Реализация функции main. Обработка выбора пользователя

С помощью конструкции switch-case в соответствии с указанным номером воспроизводим вызов соответствующей функции.

3.2 Модуль структур. structs.h

```
C structs.h > ...
1  #pragma once
2  #include <string>
3
4  using namespace std;
5
6  struct structDate {
7      int day, month, year;
8  };
9
10 struct structPharmacy {
11     int number;
12     structDate date;
13     string name;
14 };
15
16 struct structPerechNumber {
17     int number;
18     int n;
19 };
20
21 struct structPerechDate {
22     structDate date;
23     int n;
24 };
25
26 struct structPerechName {
27     string name;
28     int n;
29 };
30
```

Рисунок 4. Объявление всех структур программы

- #pragma once – гарантирует, что файл подключается только один раз при компиляции.
- structDate — структура для хранения даты (день, месяц, год).
- structPharmacy — описывает аптеку:
 - number — её номер;
 - date — дата открытия;
 - name — ФИО владельца.

– structPerechNumber, structPerechDate, structPerechName —
вспомогательные структуры для формирования перечней:

- number, date, name — критерии поиска;
- n — количество записей.

3.3 Модуль глобальных функций. functions.h и .cpp

```
C functions.h > ...
1  #pragma once
2  #include "structs.h"
3
4  int datecomp(structDate date_1, structDate date_2);
5
6  int validateInput();
```

Рисунок 5. Объявление глобальных функций

```
functions.cpp > ...
1  #include <iostream>
2  #include "functions.h"
3
4  // Операция сравнения дат
5  int datecomp(structDate date_1, structDate date_2) {
6      if (date_1.year > date_2.year) return 1;
7      if ((date_1.year == date_2.year) && (date_1.month > date_2.month)) return 1;
8      if ((date_1.year == date_2.year) && (date_1.month == date_2.month) && (date_1.day > date_2.day)) return 1;
9      return 0;
10 }
11
12 // Функция валидации входных данных
13 int validateInput() {
14     if (cin.fail()) {
15         string s;
16         cin.clear();
17         cin >> s;
18         return 0;
19     } else return 1;
20 }
21
```

Рисунок 6. Определение глобальных функций

- datecomp() – сравнивает 2 даты, при этом возвращает 1, если date_1 позже date_2, иначе 0;
- validateInput() – проверяет корректность ввода с клавиатуры.

3.4 Модуль класса classPharmacy_A. class_A.h и .cpp

```
classes > C class_A.h > ...
1  #pragma once
2  #include "../structs.h"
3
4  class classPharmacy_A {
5      protected:
6          structPharmacy *data;
7          int size;
8
9      public:
10         // Специальные методы класса
11         classPharmacy_A();
12         classPharmacy_A(classPharmacy_A &objectPharmacy);
13         ~classPharmacy_A();
14         classPharmacy_A& operator = (classPharmacy_A &objectPharmacy);
15
16         // Методы класса
17         void inputFile();
18         void outputData();
19         void outputFile();
20         void addPharmacy();
21         void deletePharmacy();
22         void sortNumber();
23         void sortName();
24         void sortDate();
25         void exitProgram();
26
27         // Дружественные функции класса
28         friend ostream& operator << (ostream &out, classPharmacy_A &objectPharmacy);
29     };
30
```

Рисунок 7. Объявление класса classPharmacy_A

Класс classPharmacy_A включает в себя массив данных типа structPharmacy и размер этого массива, а для обработки этого массива используются методы класса. Также определен конструктор и деструктор класса.

3.4.1 Специальные методы класса classPharmacy_A

```
classes > G+ class_A.cpp > ...
1  #include <iostream>
2  #include <conio.h>
3  #include <fstream>
4  #include <iomanip>
5  #include "class_A.h"
6  #include "../functions.h"
7
8  classPharmacy_A::classPharmacy_A(): data(NULL), size(0) {};
9
10 classPharmacy_A::~classPharmacy_A() {if (data!=NULL) delete [] data;}
11
12 // Конструктор копирования для класса classPharmacy_A
13 classPharmacy_A::classPharmacy_A(classPharmacy_A &objectPharmacy) {
14     size = objectPharmacy.size;
15     if (size==0) data=NULL;
16     else {
17         data = new structPharmacy[size];
18         if (data == NULL) {
19             std::cout << "нет памяти.\n";
20             std::cout << "Конструктор копирования.\n";
21             getch(); exit(1);
22         }
23         for (int i=0; i<size; i++)
24             data[i]=objectPharmacy.data[i];
25     }
26 }
27
28 // Операция присваивания для класса classPharmacy_A
29 classPharmacy_A& classPharmacy_A::operator = (classPharmacy_A &objectPharmacy) {
30     if (this == &objectPharmacy) return *this;
31     if (data != NULL) delete [] data;
32     size = objectPharmacy.size;
33     if (objectPharmacy.data == NULL) data = NULL;
34     else {
35         data = new structPharmacy [size];
36         if (data == NULL) {
37             std::cout<< "Нет памяти для копирования.\n";
38             getch(); return objectPharmacy;
39         }
40         for (int i = 0; i < size; i ++ )
41             data[i]=objectPharmacy.data[i];
42     }
43     return *this;
44 }
45
```

Рисунок 8. Определение специальных методов класса

В специальные методы класса входят такие методы:

– Конструктор без параметров (по умолчанию);

Изм.	Лист	№ докум.	Подп.	Дата

3271.202491.000 ПЗ

Лист
19

- Конструктор копирования;
- Деструктор;
- Перегруженная операция присваивания.

3.4.2 Методы класса `classPharmacy_A`

Методы этого класса включают в себя:

- `inputFile()` - Ввод исходного массива из файла;
- `outputData()` - Вывод массива в терминал;
- `outputFile()` - Вывод массива в текстовый файл;
- `addPharmacy()` - Добавление записи;
- `deletePharmacy()` - Удаление записи;
- `sortNumber()` - Сортировка массива по номеру аптеки;
- `sortName()` - Сортировка массива по ФИО владельца;
- `sortDate()` - Сортировка массива по дате открытия;
- `exitProgram()` - Выход из программы.

```

69 // 1. Ввод массива из файла
70 void classPharmacy_A::inputFile() {
71     ifstream fin;
72     string file;
73     string iniz;
74     structPharmacy pharmacy;
75     std::cout << "\nИмя входного файла: ";
76     cin >> file;
77     fin.open(file.c_str());
78
79     if (fin.fail()) {
80         std::cout << "Файл не открывается.\n";
81         getch(); return;
82     }
83     size=0;
84     if (data!=NULL) {
85         delete []data;
86         data=NULL;
87     }
88
89     while (true) {
90         fin >> pharmacy.number >> pharmacy.date.day
91         >> pharmacy.date.month >> pharmacy.date.year
92         >> pharmacy.name >> iniz;
93         if (fin.fail()) break;
94         size++;
95     }
96
97     fin.close();
98     fin.open(file.c_str());
99
100    data=new structPharmacy[size];
101    if (data==NULL) {
102        std::cout << "Нет памяти.\n";
103        fin.close();
104        std::cout << "Ввести файл не удастся.\n";
105        getch(); size=0; return;
106    }
107    for (int i = 0; i < size; i++) {
108        fin >> data[i].number >> data[i].date.day >> data[i].date.month
109        >> data[i].date.year >> data[i].name >> iniz;
110
111        data[i].name = data[i].name + " " + iniz;
112    }
113    fin.close();
114    std::cout<<"Файл введен"<<endl;
115    getch();
116 }

```

Рисунок 9. Определение метода inputFile()

```

118 // 2. Вывод массива в терминал
119 void classPharmacy_A::outputData() {
120
121     string str(64, '_');
122
123     std::cout << str + "\n";
124     std::cout << " |      |          Дата открытия          |      |" << "\n";
125     std::cout << " | № |  Номер аптеки  |          |      ФИО владельца      |" << "\n";
126     std::cout << " |      |          | День | Месяц | Год |          |" << "\n";
127     std::cout << str + "\n";
128
129     for (int i = 0; i < size; i++)
130         std::cout << " |" << setw(3) << i+1 << " |" << setw(15) << data[i].number << " |"
131         << setw(5) << data[i].date.day << " |" << setw(6) << data[i].date.month << " |" << setw(6)
132         << data[i].date.year << " |" << setw(17) << data[i].name << "\n";
133     std::cout << str << "\n";
134
135     getch();
136 }
137

```

Рисунок 10. Определение метода outputData()

```

138 // 3. Вывод массива в текстовый файл
139 void classPharmacy_A::outputFile() {
140     ofstream fout;
141     string file;
142
143     std::cout << "\nВведите имя выходного файла: ";
144     cin >> file;
145     fout.open(file.c_str());
146
147     if (fout.fail()) {
148         std::cout << "Файл не создается.\n";
149         getch(); return;
150     }
151
152     string str(64, '_');
153
154     fout << str + "\n";
155     fout << " |      |          Дата открытия          |      |" << "\n";
156     fout << " | № |  Номер аптеки  |          |      ФИО владельца      |" << "\n";
157     fout << " |      |          | День | Месяц | Год |          |" << "\n";
158     fout << str + "\n";
159
160     for (int i = 0; i < size; i++)
161         fout << " |" << setw(3) << i+1 << " |" << setw(15) << data[i].number << " |"
162         << setw(5) << data[i].date.day << " |" << setw(6) << data[i].date.month << " |" << setw(6)
163         << data[i].date.year << " |" << setw(17) << data[i].name << "\n";
164     fout << str << "\n";
165
166     fout.close();
167
168     std::cout << "Массив структур сохранен в файле.\n";
169     getch();
170 }
171

```

Рисунок 11. Определение метода outputFile()

```

172 // 4. Добавление записи
173 void classPharmacy_A::addPharmacy() {
174     structPharmacy pharmacy, *local_data;
175     string iniz;
176     local_data = new structPharmacy[size+1];
177
178     if (local_data==NULL) {
179         std::cout << "Нет памяти для новой записи";
180         std::cout << "Добавить не удастся.\n";
181         getch();
182         return;
183     }
184
185     std::cout << "Введите номер аптеки: ";
186     while (true) {
187         cin >> pharmacy.number;
188         if (validateInput() == 0)
189             std::cout << "Неверный формат номера аптеки. Введите еще раз: "; else break;
190     }
191
192     std::cout << "Введите дату открытия в числовом формате:\n";
193     std::cout << "День: ";
194     while (true) {
195         cin >> pharmacy.date.day;
196         if (validateInput() == 0 || pharmacy.date.day < 1 || pharmacy.date.day > 31)
197             std::cout << "Неверный формат дня. Введите еще раз: "; else break;
198     }
199
200     std::cout << "Месяц: ";
201     while (true) {
202         cin >> pharmacy.date.month;
203         if (validateInput() == 0 || pharmacy.date.month < 1 || pharmacy.date.month > 12)
204             std::cout << "Неверный формат месяца. Введите еще раз: "; else break;
205     }
206
207     std::cout << "Год: ";
208     while (true) {
209         cin >> pharmacy.date.year;
210         if (validateInput() == 0 || pharmacy.date.year < 1961 || pharmacy.date.year > 2025)
211             std::cout << "Неверный формат года. Введите еще раз: "; else break;
212     }
213
214     std::cout << "Введите фамилию и инициалы: ";
215     cin >> pharmacy.name >> iniz;
216     pharmacy.name = pharmacy.name + " " + iniz;
217
218     for (int i=0; i<size; i++)
219         local_data[i]=data[i];
220     local_data[size]=pharmacy;
221     size++;
222     if (data!=NULL) delete []data;
223     data=local_data;
224     std::cout << "Запись добавлена.\n";
225     getch();
226 }

```

Рисунок 12. Определение метода addPharmacy()

```

228 // 5. Удаление записи
229 void classPharmacy_A::deletePharmacy() {
230     int number, index;
231     char letter;
232     structPharmacy *local_data;
233     outputData();
234     std::cout << "Введите номер удаляемой строки: ";
235     cin >> number;
236
237     if (cin.fail()) {
238         string s;
239         cin.clear();
240         cin >> s;
241         std::cout << "Это не номер строки\n";
242         getch();
243         return;
244     }
245
246     if (number < 0 || number > size) {
247         std::cout << "Ошибка: нет такой строки.\n";
248         getch();
249         return;
250     }
251
252     index = number - 1;
253     std::cout << number << "-я строка:\n";
254     std::cout << data[index].number << " "
255     << data[index].date.day << " " << data[index].date.month << " "
256     << data[index].date.year << " " << data[index].name << endl;
257
258     std::cout << "Удалить?[Y/n] ";
259     cin >> letter;
260
261     if (letter == 'n') {
262         std::cout << "Отмена удаления строки.\n";
263         getch(); return;
264     } else if (letter != 'Y') {
265         std::cout << "Ошибка ответа на вопрос.\n";
266         getch(); return;
267     } else if (size==1) {
268         delete []data; data=NULL; size=0;
269     } else {
270         local_data = new structPharmacy[size-1];
271         if (local_data==NULL) {
272             std::cout << "Нет памяти.\n";
273             std::cout << "Удалить не удастся.\n";
274             getch(); return;
275         }
276         for (int i=0; i < number; i++)
277             local_data[i] = data[i];
278         for (index; index < size-1; index++)
279             local_data[index] = data[index+1];
280         delete []data;
281         data = local_data;
282         size--;
283     }
284     std::cout << "Запись удалена.\n";
285     getch();
286 }

```

Рисунок 13. Определение метода deletePharmacy()


```

288 // 6. Сортировка по номеру аптеки
289 void classPharmacy_A::sortNumber() {
290     int fl, count;
291     structPharmacy pharmacy;
292     count = size;
293     do {
294         fl = 0; count--;
295         for (int i = 0; i < count; i++) {
296             if (data[i].number > data[i+1].number) {
297                 fl = 1; pharmacy = data[i];
298                 data[i] = data[i+1];
299                 data[i+1] = pharmacy;
300             }
301         }
302     } while (fl==1);
303     std::cout << "Массив структур упорядочен по номеру аптеки\n";
304     getch();
305 }
306
307 // 7. Сортировка по ФИО
308 void classPharmacy_A::sortName() {
309     int fl, count;
310     structPharmacy pharmacy;
311     count = size;
312     do {
313         fl = 0; count--;
314         for (int i = 0; i < count; i++) {
315             if (data[i].name > data[i+1].name) {
316                 fl = 1; pharmacy = data[i];
317                 data[i] = data[i+1];
318                 data[i+1] = pharmacy;
319             }
320         }
321     } while (fl==1);
322     std::cout << "Массив структур упорядочен по ФИО в алфавитном порядке\n";
323     getch();
324 }
325
326 // 8. Сортировка по дате открытия
327 void classPharmacy_A::sortDate() {
328     int fl, count;
329     structPharmacy pharmacy;
330     count = size;
331     do {
332         fl = 0; count--;
333         for (int i = 0; i < count; i++) {
334             if (datecomp(data[i].date, data[i+1].date) > 0) {
335                 fl = 1; pharmacy = data[i];
336                 data[i] = data[i+1];
337                 data[i+1] = pharmacy;
338             }
339         }
340     } while (fl==1);
341     std::cout << "Массив структур упорядочен по дате открытия.\n";
342     getch();
343 }

```

Рисунок 14. Определение методов sortNumber(), sortName(), sortDate()

```

345 // 24. Выход из программы
346 void classPharmacy_A::exitProgram() {
347     char letter;
348
349     std::cout << "Завершить работу программы?[Y/n]" << endl;
350     cin >> letter;
351
352     if (letter == 'n') {
353         std::cout << "Отмена завершения.\n";
354         getch(); return;
355     } else if (letter != 'Y') {
356         std::cout << "Ошибка ответа на вопрос.\n";
357         getch(); return;
358     } else {
359         std::cout << "Программа завершила свою работу\n" << endl;
360         exit(0);
361     }
362 }
363

```

Рисунок 15. Определение методов exitProgram()

3.4.3 Дружественные функции класса classPharmacy_A

```

46 // Операция вывода массива в терминал
47 ostream& operator << (ostream &out, classPharmacy_A &objectPharmacy) {
48     string str(64, '_');
49
50     out << str + "\n";
51     out << "|   |   |   Дата открытия   |   << "\n";
52     out << "| № |   Номер аптеки   |_____| ФИО владельца |" << "\n";
53     out << "|   |   |   День | Месяц | Год   |   << "\n";
54     out << str + "\n";
55
56     for (int i = 0; i < objectPharmacy.size; i++)
57         out << "|" << setw(3) << i+1 << " |" << setw(15)
58             << objectPharmacy.data[i].number << " |" << setw(5)
59             << objectPharmacy.data[i].date.day << " |" << setw(6)
60             << objectPharmacy.data[i].date.month << " |" << setw(6)
61             << objectPharmacy.data[i].date.year << " |" << setw(17)
62             << objectPharmacy.data[i].name << "\n";
63     out << str << "\n";
64
65     getch();
66     return out;
67 }
68

```

Рисунок 16. Определение перегруженной операции вывода на экран

3.5 Модуль класса classPharmacy_B. class_B.h и .cpp

```
classes > C class_B.h > ...
1  #pragma once
2  #include "class_A.h"
3
4  class classPharmacy_B :public classPharmacy_A {
5      protected:
6          structPerechNumber *data_pnumber;
7          structPerechDate *data_pdate;
8          structPerechName *data_pname;
9          int size_pnumber, size_pdate, size_pname;
10
11     public:
12         classPharmacy_B() {
13             data_pnumber = NULL; size_pnumber = 0;
14             data_pdate = NULL; size_pdate = 0;
15             data_pname = NULL; size_pname = 0;
16         };
17         classPharmacy_B(classPharmacy_B &objectPharmacy);
18         ~classPharmacy_B() {
19             if (data_pnumber!=NULL) delete [] data_pnumber;
20             if (data_pdate!=NULL) delete [] data_pdate;
21             if (data_pname!=NULL) delete [] data_pname;
22         }
23         classPharmacy_B& operator = (classPharmacy_B &objectPharmacy);
24
25         // Методы класса
26         void outputData(string &id);
27         void outputFile(string &id);
28         void sortNumber();
29         void sortCount();
30         void sortDate();
31         void makePerech(string &id);
32
33         // Дружественные функции класса
34
35         friend string& operator << (string &id, classPharmacy_B &objectPharmacy);
36     };
37
```

Рисунок 17. Объявление класса classPharmacy_B

Класс classPharmacy_B наследует сущности класса classPharmacy_A и предназначен для формирования перечней, основанных на записях в основном массиве данных класса classPharmacy_A.

Включает в себя 3 массива данных:

- Перечень номеров аптек с типом structPerechNumber;
- Перечень ФИО владельцев с типом structPerechName;
- Перечень дат открытия с типом structPerechDate.

Имеет методы для обработки сформированного перечня.

3.5.1 Специальные методы класса classPharmacy_B

В специальные методы класса входят такие методы:

- Конструктор без параметров (по умолчанию);
- Конструктор копирования;
- Деструктор;
- Перегруженная операция присваивания.

```
8 // Конструктор копирования для класса classPharmacy_B
9 classPharmacy_B::classPharmacy_B(classPharmacy_B &objectPharmacy):classPharmacy_A(objectPharmacy) {
10     size_pnumber = objectPharmacy.size_pnumber;
11     size_pdate = objectPharmacy.size_pdate;
12     size_pname = objectPharmacy.size_pname;
13
14     if (size_pnumber==0) data_pnumber=NULL;
15     else {
16         data_pnumber = new structPerechNumber[size_pnumber];
17         if (data_pnumber == NULL) {
18             std::cout << "нет памяти.\n";
19             std::cout << "Конструктор копирования.\n";
20             getch(); exit(1);
21         }
22         for (int i=0; i<size_pnumber; i++)
23             data_pnumber[i]=objectPharmacy.data_pnumber[i];
24     }
25
26     if (size_pdate==0) data_pdate=NULL;
27     else {
28         data_pdate = new structPerechDate[size_pdate];
29         if (data_pdate == NULL) {
30             std::cout << "нет памяти.\n";
31             std::cout << "Конструктор копирования.\n";
32             getch(); exit(1);
33         }
34         for (int i=0; i<size_pdate; i++)
35             data_pdate[i]=objectPharmacy.data_pdate[i];
36     }
37
38     if (size_pname==0) data_pname=NULL;
39     else {
40         data_pname = new structPerechName[size_pname];
41         if (data_pname == NULL) {
42             std::cout << "нет памяти.\n";
43             std::cout << "Конструктор копирования.\n";
44             getch(); exit(1);
45         }
46         for (int i=0; i<size_pname; i++)
47             data_pname[i]=objectPharmacy.data_pname[i];
48     }
49 }
```

Рисунок 18. Определение конструктора копирования

```

51 // Операция присваивания для класса classPharmacy_B
52 classPharmacy_B& classPharmacy_B::operator = (classPharmacy_B &objectPharmacy) {
53     if (this == &objectPharmacy) return *this;
54     classPharmacy_A::operator = (objectPharmacy);
55
56     if (size_pnumber==0) data_pnumber=NULL;
57     else {
58         data_pnumber = new structPerechNumber[size_pnumber];
59         if (data_pnumber == NULL) {
60             std::cout << "нет памяти.\n";
61             std::cout << "Конструктор копирования.\n";
62             getch(); exit(1);
63         }
64         for (int i=0; i<size_pnumber; i++)
65             data_pnumber[i]=objectPharmacy.data_pnumber[i];
66     }
67
68     if (size_pdate==0) data_pdate=NULL;
69     else {
70         data_pdate = new structPerechDate[size_pdate];
71         if (data_pdate == NULL) {
72             std::cout << "нет памяти.\n";
73             std::cout << "Конструктор копирования.\n";
74             getch(); exit(1);
75         }
76         for (int i=0; i<size_pdate; i++)
77             data_pdate[i]=objectPharmacy.data_pdate[i];
78     }
79
80     if (size_pname==0) data_pname=NULL;
81     else {
82         data_pname = new structPerechName[size_pname];
83         if (data_pname == NULL) {
84             std::cout << "нет памяти.\n";
85             std::cout << "Конструктор копирования.\n";
86             getch(); exit(1);
87         }
88         for (int i=0; i<size_pname; i++)
89             data_pname[i]=objectPharmacy.data_pname[i];
90     }
91     return *this;
92 }

```

Рисунок 19. Определение перегруженной операции присваивания

3.5.2 Методы класса classPharmacy_B

Методы этого класса включают в себя:

- outputData(id) - Вывод одного из перечней в терминал при указании id перечня: “number”, “name” или “date” для перечней номеров аптек, дат открытия и ФИО владельцев соответственно;
- outputFile(id) - Вывод одного из перечней в текстовый файл;
- sortNumber() - Сортировка перечня номеров аптек по номеру аптеки;
- sortCount() - Сортировка перечня ФИО владельцев по количеству аптек;
- sortDate() - Сортировка перечня дат открытия по дате открытия;
- makePerech(id) – формирование перечня по id перечня.

```

307 // 9/12/15. Формирование перечня
308 void classPharmacy_B::makePerech(string &id) {
309     int flag;
310     // 9. Формирование перечня номеров аптек
311     if (id == "number") {
312         size_pnumber = 0;
313         structPerechNumber *local_data_pnumber;
314         local_data_pnumber = new structPerechNumber [size];
315         if (local_data_pnumber == NULL) {
316             std::cout << "Нет памяти.\n";
317             getch(); return;}
318
319         for (int i=0; i<size; i++) {
320             flag=0;
321             for (int j=0; j<size_pnumber; j++) {
322                 if (data[i].number == local_data_pnumber[j].number) {
323                     flag = 1;
324                     local_data_pnumber[j].n++;
325                 }
326             }
327             if (flag==0) {
328                 local_data_pnumber[size_pnumber].number = data[i].number;
329                 local_data_pnumber[size_pnumber].n = 1;
330                 size_pnumber++;
331             }
332         }
333         if (data_pnumber != NULL) delete [] data_pnumber;
334         data_pnumber = new structPerechNumber [size_pnumber];
335         if (data_pnumber == NULL) {
336             std::cout << "Нет памяти для перечня.\n";
337             getch(); size_pnumber = 0; delete [] local_data_pnumber;
338             return;
339         }
340         for (int j=0; j<size_pnumber; j++)
341             data_pnumber[j] = local_data_pnumber[j];
342         delete [] local_data_pnumber;
343
344         sortNumber();
345         std::cout << "Перечень сформирован.\n";
346         getch();

```

Рисунок 20. Определение метода makePerech(). Формирование перечня номеров аптек

```

348 // 12. Формирование перечня дат открытия
349 } else if (id == "date") {
350     size_pdate = 0;
351     structPerechDate *local_data_pdate;
352     local_data_pdate = new structPerechDate [size];
353     if (local_data_pdate == NULL) {
354         std::cout << "Нет памяти.\n";
355         getch(); return;}
356
357     for (int i=0; i<size; i++) {
358         flag=0;
359         for (int j=0; j<size_pdate; j++) {
360             if (data[i].date.day == local_data_pdate[j].date.day &&
361                 data[i].date.month == local_data_pdate[j].date.month &&
362                 data[i].date.year == local_data_pdate[j].date.year) {
363                 flag = 1;
364                 local_data_pdate[j].n++;
365             }
366         }
367         if (flag==0) {
368             local_data_pdate[size_pdate].date = data[i].date;
369             local_data_pdate[size_pdate].n = 1;
370             size_pdate++;
371         }
372     }
373     if (data_pdate != NULL) delete [] data_pdate;
374     data_pdate = new structPerechDate [size_pdate];
375     if (data_pdate == NULL) {
376         std::cout << "Нет памяти для перечня.\n";
377         getch(); size_pdate = 0; delete [] local_data_pdate;
378         return;
379     }
380     for (int j=0; j<size_pdate; j++)
381         data_pdate[j] = local_data_pdate[j];
382     delete [] local_data_pdate;
383
384     sortDate();
385     std::cout << "Перечень сформирован.\n";
386     getch();

```

Рисунок 21. Определение метода makePerech(). Формирование перечня дат открытия


```

388 // 15. Формирование перечня ФИО владельцев
389 } else if (id == "name") {
390     size_pname = 0;
391     structPerechName *local_data_pname;
392     local_data_pname = new structPerechName [size];
393     if (local_data_pname == NULL) {
394         std::cout << "Нет памяти.\n";
395         getch(); return;
396     }
397     for (int i=0; i<size; i++) {
398         flag=0;
399         for (int j=0; j<size_pname; j++) {
400             if (data[i].name == local_data_pname[j].name) {
401                 flag = 1;
402                 local_data_pname[j].n++;
403             }
404         }
405         if (flag==0) {
406             local_data_pname[size_pname].name = data[i].name;
407             local_data_pname[size_pname].n = 1;
408             size_pname++;
409         }
410     }
411     if (data_pname != NULL) delete [] data_pname;
412     data_pname = new structPerechName [size_pname];
413     if (data_pname == NULL) {
414         std::cout << "Нет памяти для перечня.\n";
415         getch(); size_pname = 0; delete [] local_data_pname;
416         return;
417     }
418     for (int j=0; j<size_pname; j++)
419         data_pname[j] = local_data_pname[j];
420     delete [] local_data_pname;
421
422     sortCount();
423     std::cout << "Перечень сформирован.\n";
424     getch();
425
426 }
427 }

```

Рисунок 22. Определение метода makePerech(). Формирование перечня ФИО владельцев

```

94 // 10/13/16. Вывод перечня в терминал
95 void classPharmacy_B::outputData(string &id) {
96
97     if (id == "number") {
98         string str(46, '_');
99         std::cout << str + "\n";
100         std::cout << "| № | Номер аптеки | Количество владельцев |" << "\n";
101         std::cout << str + "\n";
102
103         for (int i = 0; i < size_pnumber; i++)
104             std::cout << "|" << setw(3) << i+1 << " |" << setw(15)
105                 << data_pnumber[i].number << " |" << setw(22)
106                 << data_pnumber[i].n << " |" << "\n";
107         std::cout << str << "\n";
108
109         getch();
110     } else if (id == "date") {
111         string str(48, '_');
112         std::cout << str + "\n";
113         std::cout << "| | Дата открытия |" << "\n";
114         std::cout << "| № | _____ | Количество аптек |" << "\n";
115         std::cout << "| | День | Месяц | Год |" << "\n";
116         std::cout << str + "\n";
117
118         for (int i = 0; i < size_pdate; i++)
119             std::cout << "|" << setw(3) << i+1 << " |" << setw(5)
120                 << data_pdate[i].date.day << " |" << setw(6)
121                 << data_pdate[i].date.month << " |" << setw(5)
122                 << data_pdate[i].date.year << " |" << setw(20)
123                 << data_pdate[i].n << " |" << "\n";
124         std::cout << str << "\n";
125
126         getch();
127     } else if (id == "name") {
128         string str(46, '_');
129         std::cout << str + "\n";
130         std::cout << "| № | ФИО владельца | Количество аптек |" << "\n";
131         std::cout << str + "\n";
132
133         for (int i = 0; i < size_pname; i++)
134             std::cout << "|" << setw(3) << i+1 << " |" << setw(18)
135                 << data_pname[i].name << " |" << setw(19)
136                 << data_pname[i].n << " |" << "\n";
137         std::cout << str << "\n";
138
139         getch();
140     }
141 }

```

Рисунок 23. Определение метода outputData() с указанием параметра

```

143 // 11/14/17. Вывод перечня в текстовый файл
144 void classPharmacy_B::outputFile(string &id) {
145     ofstream fout;
146     string file;
147
148     std::cout << "\nВведите имя выходного файла: ";
149     cin >> file;
150     fout.open(file.c_str());
151
152     if (fout.fail()) {
153         fout << "Файл не создается.\n";
154         getch(); return;
155     }
156
157     if (id == "number") {
158         string str(46, '_');
159         fout << str + "\n";
160         fout << "| № | Номер аптеки | Количество владельцев |" << "\n";
161         fout << str + "\n";
162
163         for (int i = 0; i < size_pnumber; i++)
164             fout << "|" << setw(3) << i+1 << " |" << setw(15)
165                 << data_pnumber[i].number << " |" << setw(22)
166                 << data_pnumber[i].n << " |" << "\n";
167         fout << str << "\n";
168
169     } else if (id == "date") {
170         string str(48, '_');
171         fout << str + "\n";
172         fout << "| | Дата открытия | |" << "\n";
173         fout << "| № | | Количество аптек |" << "\n";
174         fout << "| | День | Месяц | Год | |" << "\n";
175         fout << str + "\n";
176
177         for (int i = 0; i < size_pdate; i++)
178             fout << "|" << setw(3) << i+1 << " |" << setw(5)
179                 << data_pdate[i].date.day << " |" << setw(6)
180                 << data_pdate[i].date.month << " |" << setw(5)
181                 << data_pdate[i].date.year << " |" << setw(20)
182                 << data_pdate[i].n << " |" << "\n";
183         fout << str << "\n";
184
185     } else if (id == "name") {
186         string str(46, '_');
187         fout << str + "\n";
188         fout << "| № | ФИО владельца | Количество аптек |" << "\n";
189         fout << str + "\n";
190
191         for (int i = 0; i < size_pname; i++)
192             fout << "|" << setw(3) << i+1 << " |" << setw(18)
193                 << data_pname[i].name << " |" << setw(19)
194                 << data_pname[i].n << " |" << "\n";
195         fout << str << "\n";
196     }
197     fout.close();
198     std::cout << "Перечень сохранен в файле.\n";
199     getch();
200 }

```

Рисунок 24. Определение метода outputFile() с указанием параметра

```

252 // Сортировка по номеру аптеки
253 void classPharmacy_B::sortNumber() {
254     int fl, count;
255     structPerechNumber pharmacy;
256
257     count = size_pnumber;
258     do {
259         fl = 0; count--;
260         for (int i = 0; i < count; i++) {
261             if (data_pnumber[i].number > data_pnumber[i+1].number) {
262                 fl = 1; pharmacy = data_pnumber[i];
263                 data_pnumber[i] = data_pnumber[i+1];
264                 data_pnumber[i+1] = pharmacy;
265             }
266         }
267     } while (fl==1);
268 }
269
270 // Сортировка по ФИО
271 void classPharmacy_B::sortCount() {
272     int fl, count;
273     structPerechName pharmacy;
274
275     count = size_pname;
276     do {
277         fl = 0; count--;
278         for (int i = 0; i < count; i++) {
279             if (data_pname[i].n > data_pname[i+1].n) {
280                 fl = 1; pharmacy = data_pname[i];
281                 data_pname[i] = data_pname[i+1];
282                 data_pname[i+1] = pharmacy;
283             }
284         }
285     } while (fl==1);
286 }
287
288
289 // Сортировка по дате открытия
290 void classPharmacy_B::sortDate() {
291     int fl, count;
292     structPerechDate pharmacy;
293
294     count = size_pdate;
295     do {
296         fl = 0; count--;
297         for (int i = 0; i < count; i++) {
298             if (datecomp(data_pdate[i].date, data_pdate[i+1].date) > 0) {
299                 fl = 1; pharmacy = data_pdate[i];
300                 data_pdate[i] = data_pdate[i+1];
301                 data_pdate[i+1] = pharmacy;
302             }
303         }
304     } while (fl==1);
305 }
306

```

Рисунок 25. Определение методов сортировок sortNumber(), sortCount(), sortDate()

3.5.3 Дружественные функции класса classPharmacy_B

```

202 // 10/13/16. Перегруженная операция вывода перечня в терминал для класса classPharmacy_B
203 string& operator << (string &id, classPharmacy_B &objectPharmacy) {
204
205     if (id == "number") {
206         string str(46, '_');
207         std::cout << str + "\n";
208         std::cout << "| № | Номер аптеки | Количество владельцев |" << "\n";
209         std::cout << str + "\n";
210
211         for (int i = 0; i < objectPharmacy.size_pnumber; i++)
212             std::cout << "|" << setw(3) << i+1 << " |" << setw(15)
213             << objectPharmacy.data_pnumber[i].number << " |" << setw(22)
214             << objectPharmacy.data_pnumber[i].n << " |" << "\n";
215         std::cout << str << "\n";
216
217         getch();
218     } else if (id == "date") {
219         string str(48, '_');
220         std::cout << str + "\n";
221         std::cout << "| | Дата открытия | |" << "\n";
222         std::cout << "| № | _____ | Количество аптек |" << "\n";
223         std::cout << "| | День | Месяц | Год | |" << "\n";
224         std::cout << str + "\n";
225
226         for (int i = 0; i < objectPharmacy.size_pdate; i++)
227             std::cout << "|" << setw(3) << i+1 << " |" << setw(5)
228             << objectPharmacy.data_pdate[i].date.day << " |" << setw(6)
229             << objectPharmacy.data_pdate[i].date.month << " |" << setw(5)
230             << objectPharmacy.data_pdate[i].date.year << " |" << setw(20)
231             << objectPharmacy.data_pdate[i].n << " |" << "\n";
232         std::cout << str << "\n";
233
234         getch();
235     } else if (id == "name") {
236         string str(46, '_');
237         std::cout << str + "\n";
238         std::cout << "| № | ФИО владельца | Количество аптек |" << "\n";
239         std::cout << str + "\n";
240
241         for (int i = 0; i < objectPharmacy.size_pname; i++)
242             std::cout << "|" << setw(3) << i+1 << " |" << setw(18)
243             << objectPharmacy.data_pname[i].name << " |" << setw(19)
244             << objectPharmacy.data_pname[i].n << " |" << "\n";
245         std::cout << str << "\n";
246
247         getch();
248     }
249     return id;
250 };
251

```

Рисунок 26. Определение перегруженной операции вывода перечня на экран

3.6 Модуль класса classPharmacy_C. class_C.h и .cpp

```
classes > C class_C.h > ...
1  #pragma once
2  #include "class_B.h"
3
4  class classPharmacy_C :public classPharmacy_B {
5      protected:
6          structPharmacy *data_sname;
7          structPharmacy *data_sdate;
8          int size_sname, size_sdate;
9
10     public:
11         classPharmacy_C() {
12             data_sdate = NULL; size_sdate = 0;
13             data_sname = NULL; size_sname = 0;
14         };
15         classPharmacy_C(classPharmacy_C &objectPharmacy);
16         ~classPharmacy_C() {
17             if (data_sdate!=NULL) delete [] data_sdate;
18             if (data_sname!=NULL) delete [] data_sname;
19         }
20         classPharmacy_C& operator = (classPharmacy_C &objectPharmacy);
21
22         // Методы класса
23         void outputData(string &id);
24         void outputFile(string &id);
25         void sortNumber();
26         void sortName();
27         void makeSearch(string &id);
28
29         // Дружественные функции класса
30         friend string& operator << (string &id, classPharmacy_C &objectPharmacy);
31     };
32
```

Рисунок 27. Объявление класса classPharmacy_C

Класс classPharmacy_B наследует сущности класса classPharmacy_C и предназначен для формирования перечней, основанных на записях в основном массиве данных класса classPharmacy_A, по поиску определенного значения для соответствующей характеристики.

Включает в себя 2 массива данных:

- Перечень записей с одним значением ФИО владельца;
- Перечень записей с одной датой открытия.

Имеет методы для обработки сформированных перечней.

3.6.1 Специальные методы класса classPharmacy_C

В специальные методы класса входят такие методы:

- Конструктор без параметров (по умолчанию);
- Конструктор копирования;
- Деструктор;
- Перегруженная операция присваивания.

```
classes > class_C.cpp > ...
1  #include <iostream>
2  #include <conio.h>
3  #include <fstream>
4  #include <iomanip>
5
6  #include "../functions.h"
7  #include "class_C.h"
8
9  // Конструктор копирования для класса classPharmacy_C
10 classPharmacy_C::classPharmacy_C(classPharmacy_C &objectPharmacy):classPharmacy_B(objectPharmacy) {
11     size_sdate = objectPharmacy.size_sdate;
12     size_sname = objectPharmacy.size_sname;
13
14     if (size_sdate==0) data_sdate=NULL;
15     else {
16         data_sdate = new structPharmacy[size_sdate];
17         if (data_sdate == NULL) {
18             std::cout << "нет памяти.\n";
19             std::cout << "Конструктор копирования.\n";
20             getch(); exit(1);
21         }
22         for (int i=0; i<size_sdate; i++)
23             data_sdate[i]=objectPharmacy.data_sdate[i];
24     }
25
26     if (size_sname==0) data_sname=NULL;
27     else {
28         data_sname = new structPharmacy[size_sname];
29         if (data_sname == NULL) {
30             std::cout << "нет памяти.\n";
31             std::cout << "Конструктор копирования.\n";
32             getch(); exit(1);
33         }
34         for (int i=0; i<size_sname; i++)
35             data_sname[i]=objectPharmacy.data_sname[i];
36     }
37 }
```

Рисунок 28. Определение конструктора копирования

```

39 // Операция присваивания для класса classPharmacy_B
40 classPharmacy_C& classPharmacy_C::operator = (classPharmacy_C &objectPharmacy) {
41     if (this == &objectPharmacy) return *this;
42     classPharmacy_B::operator = (objectPharmacy);
43
44     if (size_sdate==0) data_sdate=NULL;
45     else {
46         data_sdate = new structPharmacy[size_sdate];
47         if (data_sdate == NULL) {
48             std::cout << "нет памяти.\n";
49             std::cout << "Конструктор копирования.\n";
50             getch(); exit(1);
51         }
52         for (int i=0; i<size_sdate; i++)
53             data_sdate[i]=objectPharmacy.data_sdate[i];
54     }
55
56     if (size_sname==0) data_sname=NULL;
57     else {
58         data_sname = new structPharmacy[size_sname];
59         if (data_sname == NULL) {
60             std::cout << "нет памяти.\n";
61             std::cout << "Конструктор копирования.\n";
62             getch(); exit(1);
63         }
64         for (int i=0; i<size_sname; i++)
65             data_sname[i]=objectPharmacy.data_sname[i];
66     }
67     return *this;
68 }
69

```

Рисунок 29. Определение операции присваивания

3.6.2 Методы класса classPharmacy_C

Методы этого класса включают в себя:

- outputData(id) - Вывод одного из перечней в терминал при указании id перечня: “name” или “date” для перечней по поиску дат открытия и ФИО владельцев соответственно;
- outputFile(id) - Вывод одного из перечней в текстовый файл;
- sortNumber() - Сортировка перечня по поиску ФИО владельца по номеру аптеки;
- sortName() - Сортировка перечня по поиску даты открытия по ФИО владельца;
- makeSearch(id) – формирование перечня по id перечня.

```

220 // 18/21. Формирование перечня по поиску
221 void classPharmacy_C::makeSearch(string &id) {
222     // 18. Формирование перечня по поиску ФИО владельца
223     if (id == "name") {
224         string name, iniz;
225         size_sname = 0;
226         structPharmacy *local_data_sname;
227         local_data_sname = new structPharmacy [size];
228         if (local_data_sname == NULL) {
229             std::cout << "Нет памяти.\n";
230             getch(); return;}
231         cout << "Введите фамилию и инициалы для поиска: ";
232         cin >> name >> iniz;
233         name = name + " " + iniz;
234
235         for(int i=0; i<size; i++) {
236             if (data[i].name == name) {
237                 local_data_sname[size_sname]=data[i];
238                 size_sname++;
239             }
240         }
241
242         if (data_sname != NULL) delete [] data_sname;
243         data_sname = new structPharmacy [size_sname];
244         if (data_sname == NULL) {
245             cout << "Нет памяти для записей.\n";
246             getch(); size_sname = 0; delete [] local_data_sname; return;
247         }
248
249         for (int j=0; j<size_sname; j++)
250             data_sname[j]=local_data_sname[j];
251         delete [] local_data_sname;
252
253         sortNumber();
254         cout << "Массив записей по поиску ФИО владельца сформирован.\n";
255     }
256 }

```

Рисунок 30. Определение метода makeSearch(). Формирование перечня по поиску ФИО владельца

```

257 // 21. Формирование перечня по поиску даты открытия
258 if (id == "date") {
259     structDate date;
260     size_sdate = 0;
261     structPharmacy *local_data_sdate;
262     local_data_sdate = new structPharmacy [size];
263     if (local_data_sdate == NULL) {
264         std::cout << "Нет памяти.\n";
265         getch(); return;}
266
267
268     std::cout << "Введите дату открытия в числовом формате для поиска:\n";
269     std::cout << "День: ";
270     while (true) {
271         cin >> date.day;
272         if (validateInput() == 0 || date.day < 1 || date.day > 31)
273             std::cout << "Неверный формат дня. Введите еще раз: "; else break;
274     }
275
276     std::cout << "Месяц: ";
277     while (true) {
278         cin >> date.month;
279         if (validateInput() == 0 || date.month < 1 || date.month > 12)
280             std::cout << "Неверный формат месяца. Введите еще раз: "; else break;
281     }
282
283     std::cout << "Год: ";
284     while (true) {
285         cin >> date.year;
286         if (validateInput() == 0 || date.year < 1961 || date.year > 2025)
287             std::cout << "Неверный формат года. Введите еще раз: "; else break;
288     }
289
290     for(int i=0; i<size; i++)
291         if (data[i].date.day == date.day &&
292             data[i].date.month == date.month &&
293             data[i].date.year == date.year) {
294             local_data_sdate[size_sdate]=data[i];
295             size_sdate++;
296         }
297     if (data_sdate != NULL) delete [] data_sdate;
298     data_sdate = new structPharmacy [size_sdate];
299     if (data_sdate == NULL) {
300         cout << "Нет памяти для записей.\n";
301         getch(); size_sdate = 0; delete [] local_data_sdate; return;
302     }
303
304     for (int j=0; j<size_sdate; j++)
305         data_sdate[j]=local_data_sdate[j];
306     delete [] local_data_sdate;
307
308     sortName();
309     cout << "Массив записей по поиску даты открытия сформирован.\n";
310 }
311 getch();
312 }

```

Рисунок 31. Определение метода makeSearch(). Формирование перечня по поиску даты открытия

```

70 // 19/22. Вывод перечня по поиску в терминал
71 void classPharmacy_C::outputData(string &id) {
72     string str(64, '_');
73
74     cout << str + "\n";
75     cout << " |      |          Дата открытия      |" << "\n";
76     cout << " | № | Номер аптеки | _____ | ФИО владельца |" << "\n";
77     cout << " |      |          День | Месяц | Год      |" << "\n";
78     cout << str + "\n";
79
80 // 22. Вывод перечня по поиску даты открытия в терминал
81 if (id == "date") {
82     for (int i = 0; i < size_sdate; i++)
83         cout << " |" << setw(3) << i+1 << " |" << setw(15)
84             << data_sdate[i].number << " |" << setw(5)
85             << data_sdate[i].date.day << " |" << setw(6)
86             << data_sdate[i].date.month << " |" << setw(6)
87             << data_sdate[i].date.year << " |" << setw(17)
88             << data_sdate[i].name << "\n";
89     cout << str << "\n";
90
91 // 19. Вывод перечня по поиску ФИО владельца в терминал
92 } else if (id == "name") {
93     for (int i = 0; i < size_sname; i++)
94         cout << " |" << setw(3) << i+1 << " |" << setw(15)
95             << data_sname[i].number << " |" << setw(5)
96             << data_sname[i].date.day << " |" << setw(6)
97             << data_sname[i].date.month << " |" << setw(6)
98             << data_sname[i].date.year << " |" << setw(17)
99             << data_sname[i].name << "\n";
100     cout << str << "\n";
101 }
102 getch();
103 }
104

```

Рисунок 32. Определение метода outputData() с указанием параметра

```

106 // 20/23. Вывод перечня по поиску в текстовый файл
107 void classPharmacy_C::outputFile(string &id) {
108     ofstream fout;
109     string file;
110
111     std::cout << "\nВведите имя выходного файла: ";
112     cin >> file;
113     fout.open(file.c_str());
114
115     if (fout.fail()) {
116         fout << "Файл не создается.\n";
117         getch(); return;
118     }
119
120     string str(64, '_');
121
122     fout << str + "\n";
123     fout << " |      |          Дата открытия      |" << "\n";
124     fout << " | № | Номер аптеки |          | ФИО владельца |" << "\n";
125     fout << " |      |          День | Месяц | Год |          |" << "\n";
126     fout << str + "\n";
127
128     // 23. Вывод перечня по поиску даты открытия в текстовый файл
129     if (id == "date") {
130         for (int i = 0; i < size_sdate; i++)
131             fout << " |" << setw(3) << i+1 << " |" << setw(15)
132                 << data_sdate[i].number << " |" << setw(5)
133                 << data_sdate[i].date.day << " |" << setw(6)
134                 << data_sdate[i].date.month << " |" << setw(6)
135                 << data_sdate[i].date.year << " |" << setw(17)
136                 << data_sdate[i].name << "\n";
137         fout << str << "\n";
138
139     // 20. Вывод перечня по поиску ФИО владельца в текстовый файл
140     } else if (id == "name") {
141         for (int i = 0; i < size_sname; i++)
142             fout << " |" << setw(3) << i+1 << " |" << setw(15)
143                 << data_sname[i].number << " |" << setw(5)
144                 << data_sname[i].date.day << " |" << setw(6)
145                 << data_sname[i].date.month << " |" << setw(6)
146                 << data_sname[i].date.year << " |" << setw(17)
147                 << data_sname[i].name << "\n";
148         fout << str << "\n";
149     }
150     fout.close();
151
152     std::cout << "Перечень сохранен в файле.\n";
153     getch();
154 }
155

```

Рисунок 33. Определение метода outputFile() с указанием параметра

```

191 // Сортировка по номеру аптеки
192 void classPharmacy_C::sortNumber() {
193     int fl, count;
194     structPharmacy pharmacy;
195
196     count = size_sname;
197     do {
198         fl = 0; count--;
199         for (int i = 0; i < count; i++) {
200             if (data_sname[i].number > data_sname[i+1].number) {
201                 fl = 1; pharmacy = data_sname[i];
202                 data_sname[i] = data_sname[i+1];
203                 data_sname[i+1] = pharmacy;
204             }
205         }
206     } while (fl==1);
207 }
208
209 // Сортировка по ФИО
210 void classPharmacy_C::sortName() {
211     int fl, count;
212     structPharmacy pharmacy;
213
214     count = size_sdate;
215     do {
216         fl = 0; count--;
217         for (int i = 0; i < count; i++) {
218             if (data_sdate[i].name > data_sdate[i+1].name) {
219                 fl = 1; pharmacy = data_sdate[i];
220                 data_sdate[i] = data_sdate[i+1];
221                 data_sdate[i+1] = pharmacy;
222             }
223         }
224     } while (fl==1);
225 }
226

```

Рисунок 34. Определение методов сортировок sortNumber() и sortName()

3.6.3 Дружественные функции класса classPharmacy_C

```

156 // Оператор вывода массива в терминал для класса classPharmacy_B
157 string& operator << (string &id, classPharmacy_C &objectPharmacy) {
158
159     string str(64, '_');
160
161     cout << str + "\n";
162     cout << " |      |          Дата открытия          |" << "\n";
163     cout << " | № | Номер аптеки | _____ | ФИО владельца |" << "\n";
164     cout << " |      |          День | Месяц | Год |          |" << "\n";
165     cout << str + "\n";
166
167     if (id == "date") {
168         for (int i = 0; i < objectPharmacy.size_sdate; i++)
169             cout << " |" << setw(3) << i+1 << " |" << setw(15)
170                 << objectPharmacy.data_sdate[i].number << " |" << setw(5)
171                 << objectPharmacy.data_sdate[i].date.day << " |" << setw(6)
172                 << objectPharmacy.data_sdate[i].date.month << " |" << setw(6)
173                 << objectPharmacy.data_sdate[i].date.year << " |" << setw(17)
174                 << objectPharmacy.data_sdate[i].name << "\n";
175         cout << str << "\n";
176     }
177     else if (id == "name") {
178         for (int i = 0; i < objectPharmacy.size_sname; i++)
179             cout << " |" << setw(3) << i+1 << " |" << setw(15)
180                 << objectPharmacy.data_sname[i].number << " |" << setw(5)
181                 << objectPharmacy.data_sname[i].date.day << " |" << setw(6)
182                 << objectPharmacy.data_sname[i].date.month << " |" << setw(6)
183                 << objectPharmacy.data_sname[i].date.year << " |" << setw(17)
184                 << objectPharmacy.data_sname[i].name << "\n";
185         cout << str << "\n";
186     }
187     getch();
188     return id;
189 };
190

```

Рисунок 35. Определение перегруженной операции вывода перечня на экран

4 ТЕСТИРОВАНИЕ ПРОГРАММЫ

```
1. Ввод исходного массива из файла;
2. Вывод массива в терминал;
3. Вывод массива в текстовый файл;
4. Добавление записи;
5. Удаление записи;
6. Сортировка массива по номеру аптеки;
7. Сортировка массива по ФИО владельца;
8. Сортировка массива по дате открытия;
-----
9. Формирование перечня номеров аптек;
10. Вывод перечня номеров аптек в терминал;
11. Вывод перечня номеров аптек в текстовый файл;
-----
12. Формирование перечня дат открытия;
13. Вывод перечня дат открытия в терминал;
14. Вывод перечня дат открытия в текстовый файл;
-----
15. Формирование перечня ФИО владельцев;
16. Вывод перечня ФИО владельцев в терминал;
17. Вывод перечня ФИО владельцев в текстовый файл;
-----
18. Формирование перечня по поиску ФИО владельца;
19. Вывод перечня по поиску ФИО владельца в терминал;
20. Вывод перечня по поиску ФИО владельца в текстовый файл;
-----
21. Формирование перечня по поиску даты открытия;
22. Вывод перечня по поиску даты открытия в терминал;
23. Вывод перечня по поиску даты открытия в текстовый файл;
-----
24. Выход из программы;
Ваш выбор (1-24):
```

Рисунок 36. Главное меню программы

4.1 Работа с исходным массивом данных

input.txt

1	21	29	05	2005	Chernikov A.U.
2	33	01	01	2001	Pavlov E.A.
3	14	03	12	2003	Korshak E.P.
4	234	23	09	2005	Burikov A.S.
5	2	13	02	2015	Mardanov R.R.
6	1	15	08	2010	Kanavina K.A.
7	99	28	06	2002	Abramov V.A.
8	67	10	10	2016	Ivanov K.A.
9	124	02	01	2001	Budilov I.K.
10	87	25	07	2021	Vahitov R.R.
11	1	15	08	2010	Korshak E.P.
12	33	01	01	2001	Kovrigina A.A.
13	124	03	12	2003	Abramov V.A.
14	97	29	05	2005	Chernikov A.U.

Рисунок 37. Исходный массив данных

Ваш выбор (1-24): 1

Имя входного файла: input.txt

Файл введен

Рисунок 38. Работа пункта 1. Ввод исходного массива из файла

Ваш выбор (1-24): 2

№	Номер аптеки	Дата открытия			ФИО владельца
		День	Месяц	Год	
1	21	29	5	2005	Chernikov A.U.
2	33	1	1	2001	Pavlov E.A.
3	14	3	12	2003	Korshak E.P.
4	234	23	9	2005	Burikov A.S.
5	2	13	2	2015	Mardanov R.R.
6	1	15	8	2010	Kanavina K.A.
7	99	28	6	2002	Abramov V.A.
8	67	10	10	2016	Ivanov K.A.
9	124	2	1	2001	Budilov I.K.
10	87	25	7	2021	Vahitov R.R.
11	1	15	8	2010	Korshak E.P.
12	33	1	1	2001	Kovrigina A.A.
13	124	3	12	2003	Abramov V.A.
14	97	29	5	2005	Chernikov A.U.

Рисунок 39. Работа пункта 2. Вывод массива в терминал

Ваш выбор (1-24): 3

Введите имя выходного файла: output.txt
Массив структур сохранен в файле.

output.txt							
1							
2			Дата открытия				
3	№	Номер аптеки				ФИО владельца	
4			День	Месяц	Год		
5							
6	1	21	29	5	2005	Chernikov A.U.	
7	2	33	1	1	2001	Pavlov E.A.	
8	3	14	3	12	2003	Korshak E.P.	
9	4	234	23	9	2005	Burikov A.S.	
10	5	2	13	2	2015	Mardanov R.R.	
11	6	1	15	8	2010	Kanavina K.A.	
12	7	99	28	6	2002	Abramov V.A.	
13	8	67	10	10	2016	Ivanov K.A.	
14	9	124	2	1	2001	Budilov I.K.	
15	10	87	25	7	2021	Vahitov R.R.	
16	11	1	15	8	2010	Korshak E.P.	
17	12	33	1	1	2001	Kovrigina A.A.	
18	13	124	3	12	2003	Abramov V.A.	
19	14	97	29	5	2005	Chernikov A.U.	
20							

Рисунок 40. Работа пункта 3. Вывод массива в текстовый файл

Ваш выбор (1-24): 4
Введите номер аптеки: 907
Введите дату открытия в числовом формате:
День: qwerty
Неверный формат дня. Введите еще раз: 90
Неверный формат дня. Введите еще раз: 12
Месяц: 13
Неверный формат месяца. Введите еще раз: 11
Год: 2023
Введите фамилию и инициалы: Chernikov A.U.
Запись добавлена.

1. Ввод исходного массива из файла;
2. Вывод массива в терминал;
3. Вывод массива в текстовый файл;
4. Добавление записи;
5. Удаление записи;
6. Сортировка массива по номеру аптеки;
7. Сортировка массива по ФИО владельца;
8. Сортировка массива по дате открытия;

9. Формирование перечня номеров аптек;
10. Вывод перечня номеров аптек в терминал;
11. Вывод перечня номеров аптек в текстовый файл;

12. Формирование перечня дат открытия;
13. Вывод перечня дат открытия в терминал;
14. Вывод перечня дат открытия в текстовый файл;

15. Формирование перечня ФИО владельцев;
16. Вывод перечня ФИО владельцев в терминал;
17. Вывод перечня ФИО владельцев в текстовый файл;

18. Формирование перечня по поиску ФИО владельца;
19. Вывод перечня по поиску ФИО владельца в терминал;
20. Вывод перечня по поиску ФИО владельца в текстовый файл;

21. Формирование перечня по поиску даты открытия;
22. Вывод перечня по поиску даты открытия в терминал;
23. Вывод перечня по поиску даты открытия в текстовый файл;

24. Выход из программы;
Ваш выбор (1-24): 2

№	Номер аптеки	Дата открытия			ФИО владельца
		День	Месяц	Год	
1	21	29	5	2005	Chernikov A.U.
2	33	1	1	2001	Pavlov E.A.
3	14	3	12	2003	Korshak E.P.
4	234	23	9	2005	Burikov A.S.
5	2	13	2	2015	Mardanov R.R.
6	1	15	8	2010	Kanavina K.A.
7	99	28	6	2002	Abramov V.A.
8	67	10	10	2016	Ivanov K.A.
9	124	2	1	2001	Budilov I.K.
10	87	25	7	2021	Vahitov R.R.
11	1	15	8	2010	Korshak E.P.
12	33	1	1	2001	Kovrigina A.A.
13	124	3	12	2003	Abramov V.A.
14	97	29	5	2005	Chernikov A.U.
15	907	12	11	2023	Chernikov A.U.

Рисунок 41. Работа пункта 4. Добавление записи

Ваш выбор (1-24): 5

№	Номер аптеки	Дата открытия			ФИО владельца
		День	Месяц	Год	
1	21	29	5	2005	Chernikov A.U.
2	33	1	1	2001	Pavlov E.A.
3	14	3	12	2003	Korshak E.P.
4	234	23	9	2005	Burikov A.S.
5	2	13	2	2015	Mardanov R.R.
6	1	15	8	2010	Kanavina K.A.
7	99	28	6	2002	Abramov V.A.
8	67	10	10	2016	Ivanov K.A.
9	124	2	1	2001	Budilov I.K.
10	87	25	7	2021	Vahitov R.R.
11	1	15	8	2010	Korshak E.P.
12	33	1	1	2001	Kovrigina A.A.
13	124	3	12	2003	Abramov V.A.
14	97	29	5	2005	Chernikov A.U.
15	907	12	11	2023	Chernikov A.U.

Введите номер удаляемой строки: 8
8-я строка:
67 10 10 2016 Ivanov K.A.
Удалить?[Y/n] Y
Запись удалена.

Рисунок 42. Работа пункта 5. Удаление записи

Ваш выбор (1-24): 6

Массив структур упорядочен по номеру аптеки

1. Ввод исходного массива из файла;
 2. Вывод массива в терминал;
 3. Вывод массива в текстовый файл;
 4. Добавление записи;
 5. Удаление записи;
 6. Сортировка массива по номеру аптеки;
 7. Сортировка массива по ФИО владельца;
 8. Сортировка массива по дате открытия;
 -
 9. Формирование перечня номеров аптек;
 10. Вывод перечня номеров аптек в терминал;
 11. Вывод перечня номеров аптек в текстовый файл;
 -
 12. Формирование перечня дат открытия;
 13. Вывод перечня дат открытия в терминал;
 14. Вывод перечня дат открытия в текстовый файл;
 -
 15. Формирование перечня ФИО владельцев;
 16. Вывод перечня ФИО владельцев в терминал;
 17. Вывод перечня ФИО владельцев в текстовый файл;
 -
 18. Формирование перечня по поиску ФИО владельца;
 19. Вывод перечня по поиску ФИО владельца в терминал;
 20. Вывод перечня по поиску ФИО владельца в текстовый файл;
 -
 21. Формирование перечня по поиску даты открытия;
 22. Вывод перечня по поиску даты открытия в терминал;
 23. Вывод перечня по поиску даты открытия в текстовый файл;
 -
 24. Выход из программы;
- Ваш выбор (1-24): 2

№	Номер аптеки	Дата открытия			ФИО владельца
		День	Месяц	Год	
1	1	15	8	2010	Kanavina K.A.
2	1	15	8	2010	Korshak E.P.
3	2	13	2	2015	Mardanov R.R.
4	14	3	12	2003	Korshak E.P.
5	21	29	5	2005	Chernikov A.U.
6	33	1	1	2001	Pavlov E.A.
7	33	1	1	2001	Kovrigina A.A.
8	87	25	7	2021	Vahitov R.R.
9	97	29	5	2005	Chernikov A.U.
10	99	28	6	2002	Abramov V.A.
11	124	2	1	2001	Budilov I.K.
12	124	3	12	2003	Abramov V.A.
13	234	23	9	2005	Burikov A.S.
14	907	12	11	2023	Chernikov A.U.

Рисунок 43. Работа пункта 6. Сортировка массива по номеру аптеки

Ваш выбор (1-24): 7
Массив структур упорядочен по ФИО в алфавитном порядке

1. Ввод исходного массива из файла;
 2. Вывод массива в терминал;
 3. Вывод массива в текстовый файл;
 4. Добавление записи;
 5. Удаление записи;
 6. Сортировка массива по номеру аптеки;
 7. Сортировка массива по ФИО владельца;
 8. Сортировка массива по дате открытия;
 -
 9. Формирование перечня номеров аптек;
 10. Вывод перечня номеров аптек в терминал;
 11. Вывод перечня номеров аптек в текстовый файл;
 -
 12. Формирование перечня дат открытия;
 13. Вывод перечня дат открытия в терминал;
 14. Вывод перечня дат открытия в текстовый файл;
 -
 15. Формирование перечня ФИО владельцев;
 16. Вывод перечня ФИО владельцев в терминал;
 17. Вывод перечня ФИО владельцев в текстовый файл;
 -
 18. Формирование перечня по поиску ФИО владельца;
 19. Вывод перечня по поиску ФИО владельца в терминал;
 20. Вывод перечня по поиску ФИО владельца в текстовый файл;
 -
 21. Формирование перечня по поиску даты открытия;
 22. Вывод перечня по поиску даты открытия в терминал;
 23. Вывод перечня по поиску даты открытия в текстовый файл;
 -
 24. Выход из программы;
- Ваш выбор (1-24): 2

№	Номер аптеки	Дата открытия			ФИО владельца
		День	Месяц	Год	
1	99	28	6	2002	Abramov V.A.
2	124	3	12	2003	Abramov V.A.
3	124	2	1	2001	Budilov I.K.
4	234	23	9	2005	Burikov A.S.
5	21	29	5	2005	Chernikov A.U.
6	97	29	5	2005	Chernikov A.U.
7	907	12	11	2023	Chernikov A.U.
8	1	15	8	2010	Kanavina K.A.
9	1	15	8	2010	Korshak E.P.
10	14	3	12	2003	Korshak E.P.
11	33	1	1	2001	Kovrigina A.A.
12	2	13	2	2015	Mardanov R.R.
13	33	1	1	2001	Pavlov E.A.
14	87	25	7	2021	Vahitov R.R.

Рисунок 44. Работа пункта 7. Сортировка массива по ФИО владельца

Ваш выбор (1-24): 8

Массив структур упорядочен по дате открытия.

1. Ввод исходного массива из файла;
2. Вывод массива в терминал;
3. Вывод массива в текстовый файл;
4. Добавление записи;
5. Удаление записи;
6. Сортировка массива по номеру аптеки;
7. Сортировка массива по ФИО владельца;
8. Сортировка массива по дате открытия;

-
9. Формирование перечня номеров аптек;
 10. Вывод перечня номеров аптек в терминал;
 11. Вывод перечня номеров аптек в текстовый файл;

-
12. Формирование перечня дат открытия;
 13. Вывод перечня дат открытия в терминал;
 14. Вывод перечня дат открытия в текстовый файл;

-
15. Формирование перечня ФИО владельцев;
 16. Вывод перечня ФИО владельцев в терминал;
 17. Вывод перечня ФИО владельцев в текстовый файл;

-
18. Формирование перечня по поиску ФИО владельца;
 19. Вывод перечня по поиску ФИО владельца в терминал;
 20. Вывод перечня по поиску ФИО владельца в текстовый файл;

-
21. Формирование перечня по поиску даты открытия;
 22. Вывод перечня по поиску даты открытия в терминал;
 23. Вывод перечня по поиску даты открытия в текстовый файл;

-
24. Выход из программы;

Ваш выбор (1-24): 2

№	Номер аптеки	Дата открытия			ФИО владельца
		День	Месяц	Год	
1	33	1	1	2001	Kovrigina A.A.
2	33	1	1	2001	Pavlov E.A.
3	124	2	1	2001	Budilov I.K.
4	99	28	6	2002	Abramov V.A.
5	124	3	12	2003	Abramov V.A.
6	14	3	12	2003	Korshak E.P.
7	21	29	5	2005	Chernikov A.U.
8	97	29	5	2005	Chernikov A.U.
9	234	23	9	2005	Burikov A.S.
10	1	15	8	2010	Kanavina K.A.
11	1	15	8	2010	Korshak E.P.
12	2	13	2	2015	Mardanov R.R.
13	87	25	7	2021	Vahitov R.R.
14	907	12	11	2023	Chernikov A.U.

Рисунок 45. Работа пункта 8. Сортировка массива по дате открытия

4.2 Работа с перечнями данных

4.2.1 Перечень номеров аптек

Ваш выбор (1-24): 9
Перечень сформирован.

1. Ввод исходного массива из файла;
2. Вывод массива в терминал;
3. Вывод массива в текстовый файл;
4. Добавление записи;
5. Удаление записи;
6. Сортировка массива по номеру аптеки;
7. Сортировка массива по ФИО владельца;
8. Сортировка массива по дате открытия;

9. Формирование перечня номеров аптек;
10. Вывод перечня номеров аптек в терминал;
11. Вывод перечня номеров аптек в текстовый файл;

12. Формирование перечня дат открытия;
13. Вывод перечня дат открытия в терминал;
14. Вывод перечня дат открытия в текстовый файл;

15. Формирование перечня ФИО владельцев;
16. Вывод перечня ФИО владельцев в терминал;
17. Вывод перечня ФИО владельцев в текстовый файл;

18. Формирование перечня по поиску ФИО владельца;
19. Вывод перечня по поиску ФИО владельца в терминал;
20. Вывод перечня по поиску ФИО владельца в текстовый файл;

21. Формирование перечня по поиску даты открытия;
22. Вывод перечня по поиску даты открытия в терминал;
23. Вывод перечня по поиску даты открытия в текстовый файл;

24. Выход из программы;

Ваш выбор (1-24): 10

№	Номер аптеки	Количество владельцев
1	1	2
2	2	1
3	14	1
4	21	1
5	33	2
6	87	1
7	97	1
8	99	1
9	124	2
10	234	1
11	907	1

Рисунок 46. Работа пунктов 9. Формирование перечня номеров аптек и 10. Вывод перечня номеров аптек в терминал

Ваш выбор (1-24): 11

Введите имя выходного файла: output.txt
Перечень сохранен в файле.

output.txt				
1				
2	№	Номер аптеки	Количество владельцев	
3				
4	1	1	2	
5	2	2	1	
6	3	14	1	
7	4	21	1	
8	5	33	2	
9	6	87	1	
10	7	97	1	
11	8	99	1	
12	9	124	2	
13	10	234	1	
14	11	907	1	
15				
16				

Рисунок 47. Работа пункта 11. Вывод перечня номеров аптек в текстовый файл

4.2.2 Перечень дат открытия

Ваш выбор (1-24): 12

Перечень сформирован.

1. Ввод исходного массива из файла;
 2. Вывод массива в терминал;
 3. Вывод массива в текстовый файл;
 4. Добавление записи;
 5. Удаление записи;
 6. Сортировка массива по номеру аптеки;
 7. Сортировка массива по ФИО владельца;
 8. Сортировка массива по дате открытия;
 -
 9. Формирование перечня номеров аптек;
 10. Вывод перечня номеров аптек в терминал;
 11. Вывод перечня номеров аптек в текстовый файл;
 -
 12. Формирование перечня дат открытия;
 13. Вывод перечня дат открытия в терминал;
 14. Вывод перечня дат открытия в текстовый файл;
 -
 15. Формирование перечня ФИО владельцев;
 16. Вывод перечня ФИО владельцев в терминал;
 17. Вывод перечня ФИО владельцев в текстовый файл;
 -
 18. Формирование перечня по поиску ФИО владельца;
 19. Вывод перечня по поиску ФИО владельца в терминал;
 20. Вывод перечня по поиску ФИО владельца в текстовый файл;
 -
 21. Формирование перечня по поиску даты открытия;
 22. Вывод перечня по поиску даты открытия в терминал;
 23. Вывод перечня по поиску даты открытия в текстовый файл;
 -
 24. Выход из программы;
- Ваш выбор (1-24): 13

№	Дата открытия			Количество аптек
	День	Месяц	Год	
1	1	1	2001	2
2	2	1	2001	1
3	28	6	2002	1
4	3	12	2003	2
5	29	5	2005	2
6	23	9	2005	1
7	15	8	2010	2
8	13	2	2015	1
9	25	7	2021	1
10	12	11	2023	1

Рисунок 48. Работа пунктов 12. Формирование перечня дат открытия и 13. Вывод перечня дат открытия в терминал

Ваш выбор (1-24): 14

Введите имя выходного файла: output.txt
Перечень сохранен в файле.

output.txt

1					
2	Дата открытия				
3	№				Количество аптек
4		День	Месяц	Год	
5					
6	1	1	1	2001	2
7	2	2	1	2001	1
8	3	28	6	2002	1
9	4	3	12	2003	2
10	5	29	5	2005	2
11	6	23	9	2005	1
12	7	15	8	2010	2
13	8	13	2	2015	1
14	9	25	7	2021	1
15	10	12	11	2023	1
16					
17					

Рисунок 49. Работа пункта 14. Вывод перечня дат открытия в текстовый файл

4.2.3 Перечень ФИО владельцев

```
Ваш выбор (1-24): 15
Перечень сформирован.

1. Ввод исходного массива из файла;
2. Вывод массива в терминал;
3. Вывод массива в текстовый файл;
4. Добавление записи;
5. Удаление записи;
6. Сортировка массива по номеру аптеки;
7. Сортировка массива по ФИО владельца;
8. Сортировка массива по дате открытия;
-----
9. Формирование перечня номеров аптек;
10. Вывод перечня номеров аптек в терминал;
11. Вывод перечня номеров аптек в текстовый файл;
-----
12. Формирование перечня дат открытия;
13. Вывод перечня дат открытия в терминал;
14. Вывод перечня дат открытия в текстовый файл;
-----
15. Формирование перечня ФИО владельцев;
16. Вывод перечня ФИО владельцев в терминал;
17. Вывод перечня ФИО владельцев в текстовый файл;
-----
18. Формирование перечня по поиску ФИО владельца;
19. Вывод перечня по поиску ФИО владельца в терминал;
20. Вывод перечня по поиску ФИО владельца в текстовый файл;
-----
21. Формирование перечня по поиску даты открытия;
22. Вывод перечня по поиску даты открытия в терминал;
23. Вывод перечня по поиску даты открытия в текстовый файл;
-----
24. Выход из программы;
Ваш выбор (1-24): 16

| № | ФИО владельца | Количество аптек |
|---|---|---|
| 1 | Kovrigina A.A. | 1 |
| 2 | Pavlov E.A. | 1 |
| 3 | Budilov I.K. | 1 |
| 4 | Burikov A.S. | 1 |
| 5 | Kanavina K.A. | 1 |
| 6 | Mardanov R.R. | 1 |
| 7 | Vahitov R.R. | 1 |
| 8 | Abramov V.A. | 2 |
| 9 | Korshak E.P. | 2 |
| 10 | Chernikov A.U. | 3 |
```

Рисунок 50. Работа пунктов 15. Формирование перечня ФИО владельцев и 16. Вывод перечня ФИО владельцев в терминал

Ваш выбор (1-24): 17

Введите имя выходного файла: output.txt
Перечень сохранен в файле.

output.txt

1				
2	№	ФИО владельца	Количество аптек	
3				
4	1	Kovrigina A.A.	1	
5	2	Pavlov E.A.	1	
6	3	Budilov I.K.	1	
7	4	Burikov A.S.	1	
8	5	Kanavina K.A.	1	
9	6	Mardanov R.R.	1	
10	7	Vahitov R.R.	1	
11	8	Abramov V.A.	2	
12	9	Korshak E.P.	2	
13	10	Chernikov A.U.	3	
14				
15				

Рисунок 51. Работа пункта 17. Вывод перечня ФИО владельцев в текстовый файл

4.3 Работа с перечнями данных по поиску определенного значения

4.3.1 Перечень по поиску ФИО владельца

```
Ваш выбор (1-24): 18
Введите фамилию и инициалы для поиска: Chernikov A.U.
Массив записей по поиску ФИО владельца сформирован.

1. Ввод исходного массива из файла;
2. Вывод массива в терминал;
3. Вывод массива в текстовый файл;
4. Добавление записи;
5. Удаление записи;
6. Сортировка массива по номеру аптеки;
7. Сортировка массива по ФИО владельца;
8. Сортировка массива по дате открытия;
-----
9. Формирование перечня номеров аптек;
10. Вывод перечня номеров аптек в терминал;
11. Вывод перечня номеров аптек в текстовый файл;
-----
12. Формирование перечня дат открытия;
13. Вывод перечня дат открытия в терминал;
14. Вывод перечня дат открытия в текстовый файл;
-----
15. Формирование перечня ФИО владельцев;
16. Вывод перечня ФИО владельцев в терминал;
17. Вывод перечня ФИО владельцев в текстовый файл;
-----
18. Формирование перечня по поиску ФИО владельца;
19. Вывод перечня по поиску ФИО владельца в терминал;
20. Вывод перечня по поиску ФИО владельца в текстовый файл;
-----
21. Формирование перечня по поиску даты открытия;
22. Вывод перечня по поиску даты открытия в терминал;
23. Вывод перечня по поиску даты открытия в текстовый файл;
-----
24. Выход из программы;
Ваш выбор (1-24): 19
```

№	Номер аптеки	Дата открытия			ФИО владельца
		День	Месяц	Год	
1	21	29	5	2005	Chernikov A.U.
2	97	29	5	2005	Chernikov A.U.
3	907	12	11	2023	Chernikov A.U.

Рисунок 52. Работа пунктов 18. Формирование перечня по поиску ФИО владельца и 19. Вывод перечня по поиску ФИО владельца в терминал

Ваш выбор (1-24): 20

Введите имя выходного файла: output.txt
Перечень сохранен в файле.

output.txt							
1							
2			Дата открытия				
3	№	Номер аптеки				ФИО владельца	
4			День	Месяц	Год		
5							
6	1	21	29	5	2005	Chernikov A.U.	
7	2	97	29	5	2005	Chernikov A.U.	
8	3	907	12	11	2023	Chernikov A.U.	
9							
10							

Рисунок 53. Работа пункта 20. Вывод перечня по поиску ФИО владельца в текстовый файл

4.3.2 Перечень по поиску даты открытия

```

Ваш выбор (1-24): 21
Введите дату открытия в числовом формате для поиска:
День: 29
Месяц: wqerty
Неверный формат месяца. Введите еще раз: 05
Год: 2005
Массив записей по поиску даты открытия сформирован.

1. Ввод исходного массива из файла;
2. Вывод массива в терминал;
3. Вывод массива в текстовый файл;
4. Добавление записи;
5. Удаление записи;
6. Сортировка массива по номеру аптеки;
7. Сортировка массива по ФИО владельца;
8. Сортировка массива по дате открытия;
-----
9. Формирование перечня номеров аптек;
10. Вывод перечня номеров аптек в терминал;
11. Вывод перечня номеров аптек в текстовый файл;
-----
12. Формирование перечня дат открытия;
13. Вывод перечня дат открытия в терминал;
14. Вывод перечня дат открытия в текстовый файл;
-----
15. Формирование перечня ФИО владельцев;
16. Вывод перечня ФИО владельцев в терминал;
17. Вывод перечня ФИО владельцев в текстовый файл;
-----
18. Формирование перечня по поиску ФИО владельца;
19. Вывод перечня по поиску ФИО владельца в терминал;
20. Вывод перечня по поиску ФИО владельца в текстовый файл;
-----
21. Формирование перечня по поиску даты открытия;
22. Вывод перечня по поиску даты открытия в терминал;
23. Вывод перечня по поиску даты открытия в текстовый файл;
-----
24. Выход из программы;
Ваш выбор (1-24): 22

```

№	Номер аптеки	Дата открытия			ФИО владельца
		День	Месяц	Год	
1	21	29	5	2005	Chernikov A.U.
2	97	29	5	2005	Chernikov A.U.

Рисунок 54. Работа пунктов 21. Формирование перечня по поиску даты открытия и 22. Вывод перечня по поиску даты открытия в терминал

Ваш выбор (1-24): 23

Введите имя выходного файла: output.txt
Перечень сохранен в файле.

output.txt							
1							
2			Дата открытия				
3	№	Номер аптеки				ФИО владельца	
4			День	Месяц	Год		
5							
6	1	21	29	5	2005	Chernikov A.U.	
7	2	97	29	5	2005	Chernikov A.U.	
8							
9							

Рисунок 55. Работа пункта 23. Вывод перечня по поиску даты открытия в текстовый файл

4.3.3 Завершение работы программы

24. Выход из программы;
Ваш выбор (1-24): 24
Завершить работу программы?[Y/n]
Y
Программа завершила свою работу

Рисунок 56. Работа пункта 24. Выход из программы

ЗАКЛЮЧЕНИЕ

В результате выполнения курсового проекта была разработана программа на языке программирования C++, полностью отвечающая требованиям, а именно были спроектированы классы, необходимые для обработки массива данных, внутри классов были реализованы различные структуры хранения данных и все требуемые методы обработки. В главной функции программы было спроектировано меню, с помощью которого вызывались методы обработки.

Программа является многомодульной, включает в себя 5 заголовочных файлов с расширением .h и 5 файлов с расширением .cpp.

По результатам проектирования и тестирования программы можно сделать вывод, что программа работает безошибочно, с высоким уровнем надежности, при этом структура программы получилась логичной и удобной для расширения за счет декомпозиции её на модули.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Быков, А. Ю. Решение задач на языках программирования Си и Си++: методические указания / А. Ю. Быков. – Москва: МГТУ им. Н. Э. Баумана, 2017. – 248 с. – [Электронный ресурс] – URL: <https://e.lanbook.com/book/103505> (Дата обращения 08.04.25)

2. Михайличенко, Ж. В. Программирование на языке Си : учебно-методическое пособие / Ж. В. Михайличенко, М. А. Кузниченко, В. С. Янё. — 2-е изд. — Москва: ФЛИНТА, 2017. — 103 с. – [Электронный ресурс] – URL: <https://e.lanbook.com/book/97107> (Дата обращения 08.04.25)

3. Методические указания по лабораторным работам на языке программирования С++ [Электронный ресурс] – URL: <https://drive.google.com/drive/folders/1BEwpaOD3XccGxpzMINtUAe-LFh8erDIE?usp=sharing> (Дата обращения 08.04.25)