**СОЖЕРЖАНИЕ**

1. ВВЕДЕНИЕ……………………………………………………………..5
2. ПОСТАНОВКА ЗАДАЧИ……………………………………………..6
3. СТРУКТУРА ВХОДНЫХ И ВЫХОДНЫХ ДАННЫХ……………...7
4. ДИАГРАММА КЛАССОВ……………………………………………10
5. ОПИСАНИЕ КЛАССОВ………………………………………………11
6. СТРУКТУРНАЯ СХЕМА РАБОТЫ ПРОГРАММЫ………………..25
7. РАЗРАБОТКА АЛГОРИТМА………………………………………...26
8. КОД ПРОГРАММЫ…………………………………………………...29
9. РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ……………………………30
10. ЗАКЛЮЧЕНИЕ………………………………………………………..31
11. ЛИТЕРАТУРА…………………………………………………………32
12. ПРИЛОЖЕНИЕ А…………………………………………………….,32
13. ПРИЛОЖЕНИЕ Б……………………………………………………...33
14. ПРИЛОЖЕНИЕ В……………………………………………………..34
15. ПРИЛОЖЕНИЕ Г………………………………………………….…..35
16. ПРИЛОЖЕНИЕ Д……………………………………………………..36
17. ПРИЛОЖЕНИЕ Е……………………………………………………...37

**ВВЕДЕНИЕ**

В настоящее время информационные системы занимают значимое место в жизни людей. Их развитие не стояло на месте, продвигаясь в ногу со временем и деловыми потребностями человека. Сегодня функциональность и назначение информационных систем варьируются в широком диапазоне: от банальной возможности расчета зарплат до возможности анализировать и обрабатывать информацию, упрощая процесс принятия решений для управленческого персонала. В современных условиях, человек вынужден работать с гигантскими объемами информации. В связи с этим разработка программных продуктов, служащих для автоматизированного учета, весьма актуальна.

Язык программирования С++ представляет высокоуровневый компилируемый язык программирования общего назначения со статической типизацией, который подходит для создания самых различных приложений.

Унаследовав от Си богатые возможности по работе с памятью, С++ часто находит свое применение в системном программировании, в частности, при создании операционных систем, драйверов, различных утилит и антивирусов. С++ можно использовать в программах любого уровня, где важны скорость работы и производительность. Также особенно часто его используют для создания игр с богатой насыщенной визуализацией. В общем С++ - язык широкого пользования, на котором можно создавать практически любые виды программ.

**1 ПОСТАНОВКА ЗАДАЧИ**

Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Информация должна храниться в различных файлах, связанных определенным образом. В программе должно учитываться направление движения, дата, время отправления, стоимость билета. Программа должна по запросу выдавать информацию о наличии свободных мест, ближайших рейсах, о наличии рейсов в определенный день и время.

При реализации операций редактирования, добавления, удаления информации необходимо предусмотреть операцию отмены последних действий. Реализовать иерархию классов с использованием наследования (не менее 3-х уровней наследования). Разработать и использовать в программе классы контейнеров, итераторов и алгоритмов (свои и STL). Производить обработку исключительных ситуаций.

Аналогами программы является онлайн-сервисы по поиску и покупке билетов:

1. Сайт минского автовокзала «Центральный» <https://domovita.by/>
2. Онлайн-сервис по поиску и покупке автобусных билетов  Busfor <https://busfor.by/>

**2 СТРУКТУРА ВХОДНЫХ И ВЫХОДНЫХ ДАННЫХ**

В программе используется 10 текстовых файлов.

Файл National.txt. Хранит данные о городах (индекс города, название города), в которые осуществляются междугородные рейсы

|  |
| --- |
| 1 Витебск  2 Волковыск  3 Гомель |

Файл National\_data.txt. Хранит информацию о междугородных рейсах (индекс города, индекс рейса, время отправки, время в пути, время прибытия, стоимость, количество свободных мест, регулярность)

|  |
| --- |
| 1 1 0:6:35 0:1:32 0:8:7 12 30 ежедневно  1 2 0:6:49 0:1:18 0:8:7 12 27 по\_будням  1 3 0:6:50 0:1:43 0:8:33 12 25 выходные  2 1 0:7:14 0:1:19 0:8:33 12 30 ежедневно  2 2 0:7:50 0:1:43 0:9:33 12 12 ежедневно |

Файл International.txt. Хранит данные о городах (индекс города, название города), в которые осуществляются международные рейсы

|  |
| --- |
| 1 Вильнюс  2 Белосток  3 Владимир |

Файл International\_data.txt. Хранит информацию о международных рейсах (индекс города, индекс рейса, время отправки, время в пути, время прибытия, стоимость, количество свободных мест, регулярность)

|  |
| --- |
| 3 1 0:8:30 0:4:20 0:11:50 34 30 выходные  3 2 0:9:30 0:4:20 0:12:50 34 21 выходные  3 3 0:11:0 0:4:40 0:14:40 34 29 выходные  3 4 0:12:0 0:4:17 0:15:17 34 12 по\_будням  3 5 0:18:0 0:4:50 0:21:50 34 9 по\_будням |

Файл Excursions.txt. Хранит данные о заказных экскурсиях (индекс экскурсии, название экскурсии) по Беларуси

|  |
| --- |
| 1 Чернобыльская зона  2 Ювелирный завод ЗОРКА  3 Завод Белаз |

Файл Excursions\_data.txt. Хранит всю информацию о заказных экскурсиях по Беларуси (индекс экскурсии, время отправки, время в пути, время прибытия, стоимость, количество свободных мест, описание)

|  |
| --- |
| 1 0 0:7:0 0:12:0 0:19:0 30 30  Посещение музея изолированной территории заповедника Чернобыльской зоны.  У вас будет возможность наблюдать за уникальной природой.  ---  2 0 0:12:0 0:3:0 0:15:0 12 30  Вы познакомитесь с историей завода.  Познакомитесь со сложным технологическим процессом.  После просмотра экскурсии мы посетим фирменный шоу-рум завода.  --- |

Файл Tours\_in\_Belarus.txt. Хранит данные о заказных турах по Беларуси (индекс тура, название тура)

|  |
| --- |
| 1 Беловежская Пуща  2 Река Птичь. Дудутки  3 Историко-культурный комплекс Линия Сталина |

Файл Tours\_in\_Belarus\_data.txt. Хранит всю информацию о заказных турах по Беларуси (индекс тура, время отправки, время в пути, время прибытия, стоимость, количество свободных мест, описание, маршрут ура)

|  |
| --- |
| 1 0 0:9:50 0:0:25 0:10:15 3 30  Экскурсия в Беловежскую Пущу .  Посещение вольеров с животными.  Экскурсия по г. Бресту.  Посещение Брестской крепости.  ---  Музей в Брестской крепости,Поместье Деда Мороза,Музей природы,---, |

Файл Overseas\_tours.txt. Хранит данные о заграничных заказных турах (индекс тура, название тура)

|  |
| --- |
| 1 Австро-Венгерская сказка  2 Романтическая дорога Баварии  3 Студенческие каникулы в Амстердаме |

Файл Overseas\_tours \_data.txt. Хранит всю информацию о заграничных заказных турах (индекс тура, время отправки, время в пути, время прибытия, стоимость, количество свободных мест, описание, маршрут тура)

|  |
| --- |
| 1 0 0:8:20 0:0:55 0:9:15 3 30  Пешеходная экскурсия по Люблину.  Посещение уникального термального водного парка, расположенного в гротах пещеры.  Экскурсия "Королевская Буда".  Посеoщение сокровищницы Габсбургов в Хофбурге.  ---  Минск,Люблин,Будапешт,Вена,Минск,---, |

Также в ходе работы программы для хранения промежуточных данных используются STL-контейнеры:

-vector (используется для хранения текстовой информации)

-stack (используется в реализации отмены последних действий)

- написанный вручную шаблонный контейнер List, в который записываются данные из файлов. Непосредственно в самом контейнере и происходят операции удаления, добавления, изменения и сортировки с последующей записью данных в файлы.

**3 ДИАГРАММА КЛАССОВ**

Представлена в Приложении А.

**4 ОПИСАНИЕ КЛАССОВ**

**4.1 Класс DateTime – класс, предназначенный для работы с временем**

Таблица *1 –* Класс DateTime

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| days | int | | День |
| hours | int | | Месяц |
| minutes | int | | Минуты |
| Public методы класса | | | |
| 55 | Принимаемые параметры | Возвращаемое значение | Описание |
| GetHours | − | int | Возвращает количество часов |
| GetMinutes | − | void | Возвращает количество минут |
| Print | ostream& out | void | Вывод объекта класса DateTime на экран в формате дни:часы:минуты |
| Input | istream& in | istream& | Чтение объекта класса DateTime из передаваемого потока |
| Record | ostream& out | ostream& | Запись объекта класса DateTime в файл через передаваемый поток |
| operator== | DateTime& obj | bool | Сравнение двух объектов класса DateTime |
| operator= | DateTime& obj | DateTime | Присваивание |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| operator<< | ostream& out, DateTime& obj | ostream& | Вывод объекта класса DateTime на экран |
| operator> | DateTime& obj | bool | Сравнение по времени |
| operator>> | istream& in, DateTime& obj | istream& | Инициализация через консоль |
| operator- | DateTime& obj1, DateTime& obj2 | DateTime | Разность двух объектов класса DateTime |

Продолжение таблицы *1*

**4.2 Класс Route – родительский класс, содержащий общую информацию о маршрутах**

Таблица *2 –* Класс Route

|  |  |  |  |
| --- | --- | --- | --- |
| Protected поля класса | | | |
| Имя | Тип | | Описание |
| KeyId | int | | Индекс маршрута |
| routeId | int | | Индекс рейса |
| shipmentTime | DateTime | | Время отправки |
| duration | DateTime | | Продолжительность |
| arrivalTime | DateTime | | Время прибытия |
| cost | int | | Стоимость |
| seatsNumber | int | | Количество свободных мест |
| viza | bool | | Наличие визы |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| SetKeyId | int id | void | Устанавливает индекс маршрута |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| SetRouteId | int id | void | Устанавливает индекс рейса |
| GetKeyId | − | int | Возвращает индекс маршрута |
| GetRouteId | − | int | Возвращает индекс рейса |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| GetShipmentTime | void | DateTime | Возвращение времени отправки |
| Changing | − | void | Изменение информации о маршруте |
| Input | istream& in | void | Чтение объекта класса Route из передаваемого потока |
| operator= | Route& obj | Route& | Присваивание |
| operator== | Route& obj | bool | Сравнение двух объектов класса Route |
| Record | ostream& out | ostream& | Запись объекта класса Route в файл через передаваемый поток |
| operator<< | ostream& out, Route& r | ostream& | Вывод объекта класса Route на экран |
| operator>> | istream& in, Rpute& r | istream& | Инициализация через консоль |

Продолжение таблицы *2*

**4.3 Класс Trip – класс междугородных и международных рейсов**

Таблица *3 –* Класс Trip

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| arrivalPoint | std::string | | Место прибытия |
| Имя | Тип | | Описание |
| regularity | std::string | | Регулярность |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| GetReg | − | std::string | Возвращает регулярность |
| Input | istream& in | istream& | Чтение объекта класса Trip из передаваемого потока |
| Record | ostream& out | ostream& | Запись объекта класса Trip в файл через передаваемый поток |
| operator= | Trip& obj | Trip& | Присваивание |
| operator== | Trip& obj | bool | Сравнение двух объектов класса Trip |
| Changing | − | void | Изменение информации о рейсе |
| operator>> | istream& in, Trip& obj | istream& | Инициализация через консоль |
| operator<< | ostream& out, Trip& obj | ostream& | Вывод объекта класса Trip на экран |

**4.4 Класс Excursion – класс экскурсий**

Таблица *4 –* Класс Excursion

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| title | std::string | | Название |
| Имя | Тип | | Описание |
| description | vector<std::string> | | Описание экскурсии |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| SetTitle | std::string& str | void | Устанавливает название |
| Input | istream& in | istream& | Чтение объекта класса Excursion из передаваемого потока |
| Record | ostream& out | ostream& out | Запись объекта класса Excursion в файл через передаваемый поток |
| Changing | − | void | Изменение информации об экскурсии |
| operator== | Excursion& obj | bool | Сравнение двух объектов класса Excursion по названию |
| operator>> | istream& in, Excursion& obj | istream& | Инициализация через консоль |
| operator<< | ostream& out, Excursion& obj | ostream& | Вывод объекта класса Excursion на экран |

Продолжение таблицы *4*

**4.5 Класс Tour – класс туров**

Таблица *5 –* Класс Tour

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| tourRoad | vector<std::string> | | маршрут тура |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| Input | istream& in | istream& | Чтение объекта класса Tour из передаваемого потока |
| Record | ostream& out | ostream& | Запись объекта класса Tour в файл через передаваемый поток |
| Changing | − | void | Изменение информации о туре |
| operator>> | istream& in, Tour& obj | istream& | Инициализация через консоль |
| operator<< | ostream& out, Tour& obj | ostream& | Вывод объекта класса Tour на экран |

**4.6 Класс Key – класс, работающий с ключевыми объектами**

Таблица *6 –* Класс Key

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| id | int | | Индекс |
| name | std::string | | Название (города/экскурсии/тура) |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| SetKeyId | int newId | void | Устанавливает индекс |
| SetName | std::string | void | Устанавливает название |
| GetKeiId | − | int | Возвращает индекс |
| operator= | Key& obj | Key& | Присваивание |
| operator== | Key& obj | bool | Сравнение двух объектов класса Trip по названию |
| Input | istream& in | istream& | Чтение объекта класса Key из передаваемого потока |
| Record | ostream& out | ostream& | Запись объекта класса Key в файл через передаваемый поток |
| operator>> | istream& in, Key& obj | istream& | Инициализация через консоль |
| operator<< | ostream& out, Key& obj | ostream& | Вывод объекта класса Tour на экран |

Продолжение таблицы *6*

Продолжение таблицы *6*

**4.7 Структура TeamStack, предназначенная для осуществления операции отмены последних действий**

Таблица *7 –* поля структуры TeamStack

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Описание |
| operationStack | std::stack<int> | Сохранение типа выбранной пользователем операции |
| optionStack | std::stack<int> | Сохранение типа объекта, подвергающегося выбранной пользователем операции |
| Имя | Принимаемые параметры | Возвращаемое значение |
| keyStack | std::stack<Key> | Сохранение объекта класса Key |
| objectStack | std::stack<std::shared\_ptr<Route>> | Сохранение объектов дочерних классов от класса Route |
| Имя | Тип | Описание |
| savedList | std::vector<Trip> | Сохранение всех рейсов конкретного города в случае отмены операции его удаления |

Продолжение таблицы *7*

**4.8 Шаблонный класс TextFile, предназначенный для хранения данных различных типов**

Таблица *8 –* класс TextFile

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| filePath | std::string | | Путь к файлу |
| fileStream | fsream | | Файловый поток |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| GetFileStream | − | fstream& | Возвращает файловый поток |
| GetFilePath | − | std::string | Возвращает путь к файлу |
| SetFlePath | std::string str | void | Устанавливает путь к файлу |
| IsOpen | − | bool | Проверка на открытие файла |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| Open | unsigned int mode | bool | Открытие файла |
| Close | − | void | Закрытие файла |
| RecordInFile | T& obj | bool | Запись объекта в конец файла |
| GetLastKeyIndex | − | int | Получение индекса последнего элемента в файле |
| Print | − | void | Вывод содержимого файла на экран |
| Find | T& obj | bool | Поиск конкретного объекта |
| GetContentByIndex | int i | T | Возвращается объект по заданному индексу |
| PrintByIndex | int i | void | Вывод объектов по индексу (вывод всех рейсов конкретного города) |
| PrintByRequest | int i, std::string date | void | Вывод объектов по заданному запросу (рейсов по дате) |
| Clear | − | void | Очистка списка |

Продолжение таблицы *8*

**4.9 Шаблонная структура listNode, предназначенная для работы с шаблонным классом-контейнером List**

Таблица *9 –* поля структуры listNode

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Описание |
| value | T | Значение узла |
| next | listNode<T>\* | Указатель на следующий узел |
| Имя | Тип | Описание |
| prev | listNode<T>\* | Указатель на предыдущий узел |

Продолжение таблицы *9*

**4.10 Шаблонный класс List, предназначенный для хранения данных различных типов**

Таблица *10 –* класс List

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| listHead | listNode<T>\* | | Указатель на первый элемент списка |
| listCurrent | listNode<T>\* | | Указатель на последний элемент списка |
| listSize | int | | Длина списка |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| IsClear | − | bool | Проверка списка на пустоту |
| Clear | − | void | Очистка списка |
| Add | T& obj | void | Добавление нового элемента в конец списка |
| Delete | T& obj | void | Удаление элемента списка |
| Print | − | void | Вывод содержимого списка на экран |
| FillFromFile | T1& file | void | Заполнение списка из файла |
| WriteToFile | T1& file, int option | void | Запись всего содержимого списка в файл |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| TripsInsertion | T1& file, T obj, bool option | void | Отсортированная вставка/удаление объекта класса Trip в список |
| Sort | − | void | Сортировка списка по индексам |
| Inbexing | − | void | Индексация элементов списка |
| begin | − | Iterator | Возвращение указателя на первый элемент списка |
| end | − | Iterator | Возвращение указателя на последний элемент списка |

Продолжение таблицы *10*

**4.11 Класс Iterator – вложенный класс, предназначенный для работы с шаблонным классом-контейнером List**

Таблица *11 –* класс Iterator

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| listLink | listNode<T>\* | | Указатель на элемент списка |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| operator++ | − | void | Переход к следующему элементу списка |
| operator-- | − | void | Переход к предыдущему элементу списка |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| operator= | Iterator& it | Iterator& | Присваивание |
| operator> | Iterator& it | bool | Сравнение двух объектов класса Iterator |
| operator== | Iterator& it | bool | Проверка на двух объектов класса Iterator на равенство |
| operator!= | Iterator& it | bool | Проверка на двух объектов класса Iterator на неравенство |
| operator\* | void | T& | Осуществляет доступ к адресуемому объекту в узле списка |

Продолжение таблицы *11*

**4.12 Класс Exceptions – родительский класс обработки исключений**

Таблица *12 –* класс Exceptions

|  |  |  |  |
| --- | --- | --- | --- |
| Protected поля класса | | | |
| Имя | Тип | | Описание |
| errorCode | inr | | Код ошибки |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| What | − | void | Возвращает суть ошибки |

**4.13 Класс BadInput – класс обработки исключений при вводе данны**

Таблица *13 –* класс BadInput

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| errorCode | int | | Код ошибки |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| What | − | void | Возвращает текс ошибки |

**4.14 Класс BadFile – класс обработки исключений при работе с файлами**

Таблица *14 –* класс BadFile

|  |  |  |  |
| --- | --- | --- | --- |
| Private поля класса | | | |
| Имя | Тип | | Описание |
| errorCode | inr | | Код ошибки |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| What | − | void | Возвращает текст ошибки |

**4.15 Шаблонный класс Algorithms, предназначенный для работы с контейнером List**

Таблица *15 –* класс Algorithms

|  |  |  |  |
| --- | --- | --- | --- |
| Public методы класса | | | |
| Имя | Принимаемые параметры | Возвращаемое значение | Описание |
| Show | Iterator begin, Iterator end | void | Вывод содержимого контейнера на экран |
| Sort | Iterator first, Iterator last | void | Сортировка содержимого контейнера |

**5 СТРУКТУРНАЯ СХЕМА РАБОТЫ ПРОГРАММЫ**

Представлена в приложении Б.

**6 РАЗРАБОТКА АЛГОРИТМА**

В данном разделе рассмотрены описания алгоритмов и схемы алгоритмов, используемые в программе.

**6.1 Схема алгоритмов**

Схема алгоритма метода PrintByRequest(int, string) и схема алгоритма записи данных в список из файла представлены в Приложении В.

**6.2 Алгоритмы по шагам**

**6.2.1 Алгоритм поиска объектов в файле по определённым параметрам**

Рассмотрим метод PrintByRequest поиска и вывода данных на экран по определённым параметрам.

1. Начало.
2. Открытие файла. Если при открытии произошла ошибка, то генерируется исключение: объект класса BadFile, на экран выводится текст ошибки и завершается работа метода.
3. Входные данные: переменные int i и string date

Выходные данные: отсутствуют

Промежуточные данные: переменные bool flag для сигнализации о наличии нужных объектов в файле, int i для подсчета количества и индексации найденных объектов, объект T obj для его считывания и итерирования по файлу.

1. Цикл с условием obj.Input(fileStream) – пока объект будет считываться из файла.
2. Сравнение параметров считанного объекта с запрошенными, используя функцию RegComp(obj.GetReg(), date) и сравнение индексов. Если функция возвращает значение true и индекс объекта равен запрашиваемому i == obj.GetKeyId(), то этот объект заносится в таблицу на экране, flag переходит в значение true, счетчик найденных объектов увеличивается и печатается перегородка в таблице.
3. Конец цикла.
4. Если в цикле не было найдено необходимых объектов, то консоль очищается и на экран выводится сообщение об их отсутствии нужных объектов.
5. Закрытие файла.
6. Конец.

**6.2.2 Алгоритм записи данных в список из файла**

Рассмотрим метод FillFromFile.

1. Начало.
2. Входные данные: TextFile<T>& file – файл, из которого будут браться данные

Выходные данные: отсутствуют

Промежуточные данные: объект T obj для его считывания и итерирования по файлу.

1. Проверка списка на пустоту. Если в список уже занесены какие-либо данные, то он очищается методом Clear().
2. Открытие файла через метод file.FileOpen(ios::in).
3. Если при открытии файла произошла ошибка, то генерируется соответствующее исключение класса BadFile и прекращается работа метода.
4. Цикл с условием obj.Input(GetfileStream()) – пока объект будет считываться из файла.
5. Если считанный из файла объект не является нулевым, то он добавляется в список через метод Add(obj), иначе работа цикла прекращается.
6. Конец цикла.
7. Закрытие файла.
8. Конец.

**7 КОД ПРОГРАММЫ**

Представлен в Приложении Г.

**8 РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ**

Представлен в Приложении Д.

**ЗАКЛЮЧЕНИЕ**

В результате выполнения курсового проекта была разработана программа, представляющая из себя информационную систему автовокзала для удобной визуализации возможных маршрутов и обеспечения пользователю полноценного доступа к нужной ему информации. В программе реализованы операции добавления, удаления, редактирования информации, отмены последнего действия.

Были закреплены знания в области ООП. Была использована среда разработки Microsoft Visual Studio 2017 и операционная система Windows 10.

В дальнейшем планируется усовершенствование программы путём добавления удобного графического интерфейса и новых функций.

Системные требования:

-Операционная система Windows 98, XP, Vista,7, 8.1, 10;

-Процессор Intel Pentium III 1 GHz;

-Оперативная память 500 MB;

-Свободное место на жестком диске: 5 Мб

**ЛИТЕРАТУРА**

[1] Луцик, Ю. А. Объектно-ориентированное программирование на языке С++ : учеб. пособие / Ю. А. Луцик, В. Н. Комличенко. – Минск : БГУИР, 2008. – 266 с. : ил.

[2] Страуструп,Б. Программирование. Принципы и практика использования C++/ Б.Страуструп, Вильямс, 2018 г.

[3] Мейерс, С. Эффективный и современный С++: 42 рекомендации по использованию C++11 и C++14/C. Мейерс, Вильямс, 2016. — 304 с.

[4] Прата, С. Язык программирования C. Лекции и упражнения/С. Прата, Вильямс, 2017. — 1248 с.

**ПРИЛОЖЕНИЕ А**

**(диаграмма классов)**

**ПРИЛОЖЕНИЕ Б**

**(структурная схема)**

**ПРИЛОЖЕНИЕ В**

**(схема алгоритмов)**

**ПРИЛОЖЕНИЕ Г**

**(код программы)**

**ПРИЛОЖЕНИЕ Д**

**(результат работы программы)**

**ПРИЛОЖЕНИЕ Е**

**(ведомость документов)**