

МИНОБРНАУКИ РОССИИ
Санкт-Петербургский государственный
электротехнический университет «ЛЭТИ»
им. В.И. Ульянова (Ленина)
Кафедра САПР

КУРСОВАЯ РАБОТА
по дисциплине «Программирование»
Тема: Разработка электронной картотеки

Студенты гр. 9892

Лескин К.А.

Миллер В.В.

Преподаватель

Кузьмин С.А.

Санкт-Петербург
2020

Задание на курсовую работу

Студенты Лескин К.А., Миллер В.В.

Группа 9892

Тема работы: Обработка текстовой информации

Исходные данные:

Разработать программу, позволяющую выполнять различные операции над базой данных, представленной в виде линейного списка (тема базы данных и набор операций есть в своём варианте задания).

Курсовая работа «собирается» студентом из функций, объединённых с помощью меню в головной программе, выполняющей обработку связанного линейного списка.

В виде отдельных пользовательских функций оформляются части программы, реализующие операции:

- создание списка (выделение памяти, создание и заполнение вводимыми с клавиатуры данными элементами списка);
- сохранение введённой информации в заданном пользователем файле;
- восстановление списка (заполнение его информацией, считываемой из файла, поиск элемента по признаку (признак - одно из полей структуры));
- сортировка найденных элементов и вывод информации о них на экран;
- корректировка полей записи выбранного элемента (идентификация элемента по номеру в выводимом на экран перечне (по номеру указателя на элемент));
- удаление выбранного элемента (одного из найденных по признаку);
- вставка нового элемента (после/перед выбранным).

Индивидуальный вариант задания №10:

Обеспечить работу компании, осуществляющей грузовые перевозки на основе наличия:

- списка парка грузовиков (марка, грузоподъёмность, максимальная дальность перевозки, плановый пробег в пути за сутки);
- списка водителей (ФИО, разрешение на использование марки грузовика);
- списка маршрутов перевозки (конечный пункт, дальность, время погрузки/разгрузки в конечных пунктах, количество водителей).

При поступлении очередного заказа (маршрут, дата выезда, масса груза, пожелание по марке грузовика) необходимо сформировать для поездки комбинацию грузовик-водитель(-и).

Дополнительно необходимо выдавать информацию:

- о свободных водителях на определённую дату;
- о свободных грузовиках на определённую дату;
- о грузовиках, находящихся на определённом маршруте;
- о водителях, находящихся на определённом маршруте;
- о плановой дате прибытия грузовика с водителем(-ями).

Содержание пояснительной записки:
Требуемые разделы пояснительной записки: «Содержание», «Введение»,
Основные главы, «Заключение», «Список использованных источников».

Предполагаемый объем пояснительной записки:
Не менее 30 страниц.

Дата выдачи задания:

Дата сдачи курсовой работы:

Дата защиты курсовой работы:

Студенты гр. 9892	_____	Лескин К.А.
	_____	Миллер В.В.
Преподаватель	_____	Кузьмин С.А.

Аннотация

В данной курсовой работе необходимо реализовать программу, обрабатывающую текстовые данные, хранящиеся в виде файлов, в соответствии с опцией, которую выбирает пользователь. Данные необходимо хранить в памяти компьютера в виде списка. Пользовательский интерфейс включает в себя меню, которое позволяет производить все необходимые манипуляции с исходными данными. Для выполнения данной курсовой работы будут использованы знания, полученные при выполнении лабораторных работ. В результате будет получена программа полностью соответствующая требованиям к индивидуальному заданию.

Summary

In this coursework, it is needed to implement a program that processes text data stored as files, in accordance with the option that the user chooses. Data should be stored as a list. The user interface includes a menu that allows you to make all the necessary manipulations with the original data. In this coursework we will use the knowledge gained during laboratory work. As a result, the program will be fully compliant with the requirements for the individual task.

Содержание

1	Введение	7
2	Внешние форматы хранения данных	8
3	Внутренние форматы хранения данных	12
3.1	Структура данных двусвязный список	13
3.2	TruckBrand	13
3.3	Destination	13
3.4	Truck	14
3.5	TruckList	15
3.6	TruckDataBase	15
3.7	Route	16
3.8	RouteList	16
3.9	RouteDataBase	17
3.10	Driver	17
3.11	DriverList	18
3.12	DriverDataBase	19
3.13	Delivery	19
3.14	Schedule	20
3.15	ScheduleDataBase	20
3.16	DataBases	21
3.17	AdministratorConsole	21
3.18	Date	22
3.19	Request	23
3.20	RequestHandler	24
4	Описание пользовательских функций	25
	Список литературы	30

1 Введение

Целью данной курсовой работы является приобретение навыков разработки и отладки многомодульных программ на языке C++. Разработка программы, позволяющей создать, обработать, вывести и удалить базу данных на основе линейных списков.

В ходе выполнения работы мы воспользуемся всеми знаниями, полученными при выполнении лабораторных работ. Содержание пояснительной записки к курсовой работе соответствует стандартным требованиям к пояснительным документам для программного продукта.

В данной курсовой работе предполагается использование функций для работы с структурой данных "список"[[ссылка](#)].

2 Внешние форматы хранения данных

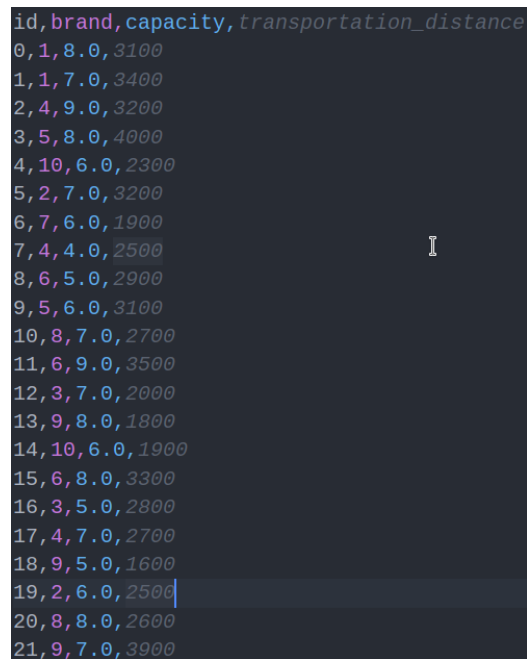
Файлы для баз данных хранятся в формате CSV (Comma Separated Values) [ссылка]. В каждом файле присутствует заголовочная строка, показывающая какие данные хранятся в каждом столбце.

CSV формат так же применяется для запросов.

Для грузовиков определены следующие столбцы:

- id – идентификационный номер грузовика (int);
- brand – код марки грузовика (int);
- capacity – грузоподъемность грузовика в тоннах (float);
- transportation_distance – максимальная дальность перевозки (int).

Пример таблицы грузовиков указан на рис. 2.1.



```
id,brand,capacity,transportation_distance
0,1,8.0,3100
1,1,7.0,3400
2,4,9.0,3200
3,5,8.0,4000
4,10,6.0,2300
5,2,7.0,3200
6,7,6.0,1900
7,4,4.0,2500
8,6,5.0,2900
9,5,6.0,3100
10,8,7.0,2700
11,6,9.0,3500
12,3,7.0,2000
13,9,8.0,1800
14,10,6.0,1900
15,6,8.0,3300
16,3,5.0,2800
17,4,7.0,2700
18,9,5.0,1600
19,2,6.0,2500
20,8,8.0,2600
21,9,7.0,3900
```

Рис. 2.1 – Пример таблицы грузовиков

Для водителей определены следующие столбцы:

- id – идентификационный номер водителя (int);
- name – имя водителя (string);
- name – фамилия водителя (string);
- name – отчество водителя (string);
- brand_code – код разрешённой марки грузовика (int).

Пример таблицы водителей указан на рис. 2.2.

```
id,name,surname,patronymic,brand_code
0,Philip,Lofin,Ivanovich,9
1,Ivan,Grachov,Sergeevich,3
2,Sergey,Pahomov,Kirillovich,1
3,Nikita,Shukin,Maksimovich,5
4,Alexey,Avdeev,Gennadievich,3
5,Denis,Spicin,Yriievich,8
6,Boris,Bobrov,Antonovich,9
7,Gregory,Vlasov,Pavlovich,10
8,Augustin,Didko,Vladimirovich,7
9,Maksim,Dobrov,Yakobovich,2
10,Konstantin,Prashin,Alexeevich,7
11,Pavel,Semechkov,Andreevich,6
12,Artem,Shilov,Dmitriievich,8
13,Seraphim,Glavin,Ignatievich,5
14,Luka,Grebovoi,Pavlovich,2
15,Vladimir,Lobov,Alexandrovich,4
16,Nikita,Shustrov,Grigorievich,4
17,Alexey,Shepchalo,Maksimovich,10
18,Daniil,Kunicin,Seraphimovich,1
19,Vyacheslav,Serbov,Semenovich,10
20,Yaroslav,Antonov,Petrovich,7
21,Viktoria,Borovaya,Alexandrovna,3
22,Svyatoslav,Kolesnikov,Pavlovich,5
23,Alexey,Shtorkin,Mihailovich,7
24,Gavriil,Bledny,Leonidovich,2
25,Pavel,Sartavalo,Sergeevich,9
26,Yrii,Zhozhin,Borisovich,6
27,Vladimir,Sheptalo,Kirillovich,4
```

Рис. 2.2 – Пример таблицы водителей

Для маршрутов определены следующие столбцы:

- id – идентификационный номер маршрута (int);
- destination_code – код конечного пункта (int);
- distance – длина маршрута в км (int);
- loading_time – время погрузки/разгрузки в конечных пунктах в ч (int);
- drivers – кол-во водителей (int);
- time_in_transit – расчётное время поставки в одну сторону (ч) (int).

Пример таблицы маршрутов указан на рис. 2.3.

```
id,destination_code,distance,loading_time,drivers,time_in_transit
0,1,2743,4,3,60
1,2,1281,2,2,30
2,3,712,1,1,22
3,4,1150,3,1,30
4,5,1622,3,2,36
5,6,1940,2,2,45
```

Рис. 2.3 – Пример таблицы маршрутов

Для графика поставок определены следующие столбцы:

- start – время отправки поставки в секундах от 1 янв 1970 00:00 (int);
- end – время возврата грузовика в секундах от 1 янв 1970 00:00 (int);
- truck_id – идентификационный номер грузовика (int);
- drivers_ids – идентификационные номера водителей (указываются через пробел)(int).

Пример таблицы графика поставок указан на рис. 2.4.

```
start,end,truck_id,drivers_ids
1606730400,1606892400,6,8
1606730400,1606957200,10,5
1606730400,1606957200,20,12
```

Рис. 2.4 – Пример таблицы графика поставок

Для запросов определены следующие столбцы:

- destination – код конечного пункта (int);
- departure_date – дата отправки в формате дд.мм.гггг чч:мм (string);
- cargo_weight – масса груза (float);
- truck_brand – код предпочитаемой марки грузовика (int).

Пример запроса указан на рис. 2.5.

```
destination,departure_date,cargo_weight,truck_brand
4,30.11.2020 13:00,4,8
```

Рис. 2.5 – Пример запроса

Для хранения пароля для доступа к консоли администратора используется обычный текстовый файл, содержащий единственную строку: хэш сумма заданного пароля.

Пример файла с хэшем пароля указан на рис. 2.6.

```
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

Рис. 2.6 – Пример файла с хэшем пароля

3 Внутренние форматы хранения данных

В данной работе используется комбинация из различных структур данных, объединённых в единую систему. Полная схема структур и их связей представлена на рис. 3.1

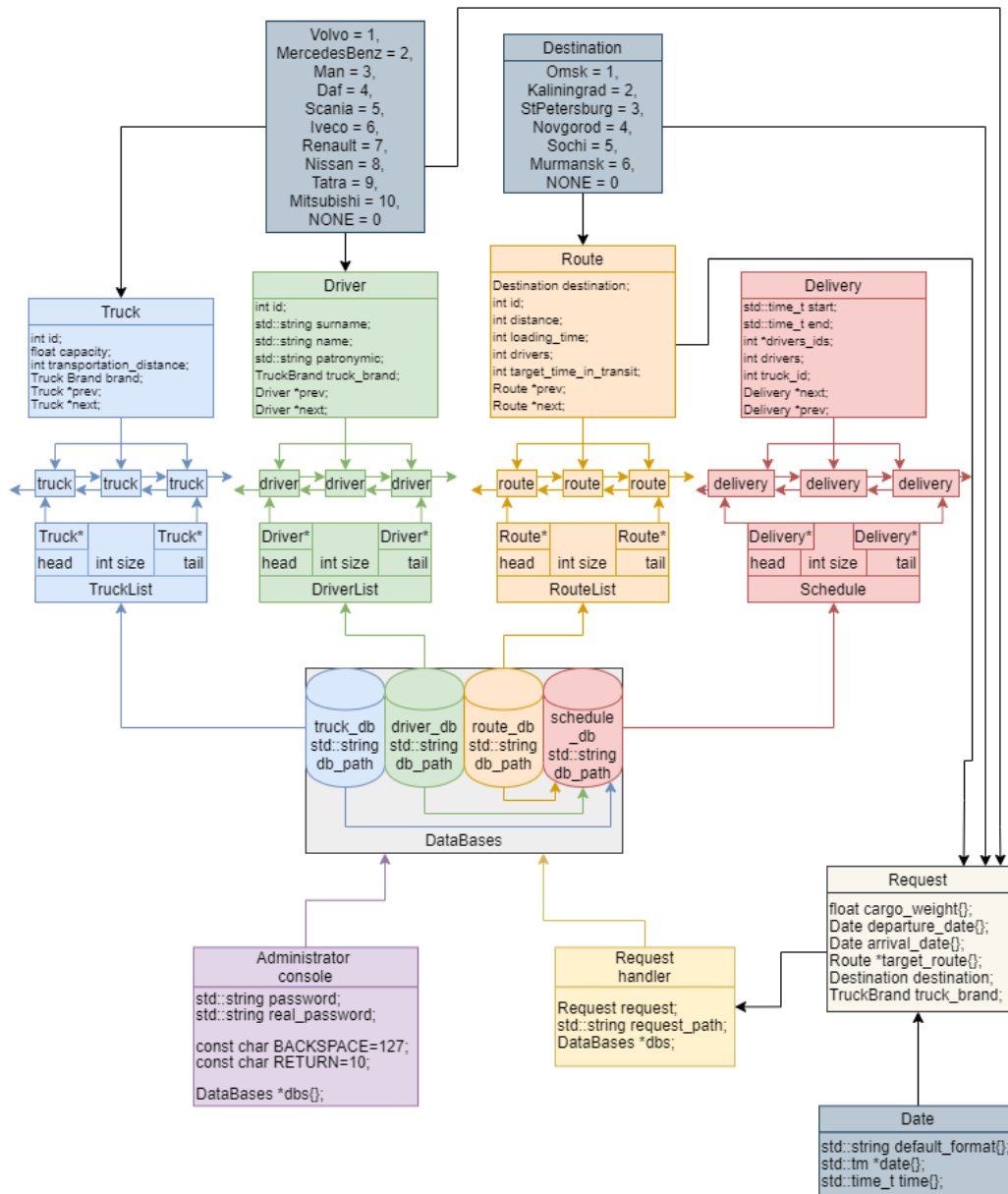


Рис. 3.1 – Обзорная диаграмма структур данных программы

3.1 Структура данных двусвязный список

В работе предполагается использование структуры данных "список". Список — структура данных, состоящая из элементов, содержащих помимо собственных данных ссылки на следующий и/или предыдущий элемент списка [2].

Для работы был выбран вариант с двунаправленным списком, где каждый узел имеет указатель на следующий и на предыдущий элементы. Схема двунаправленного списка показана на рис. 3.2.

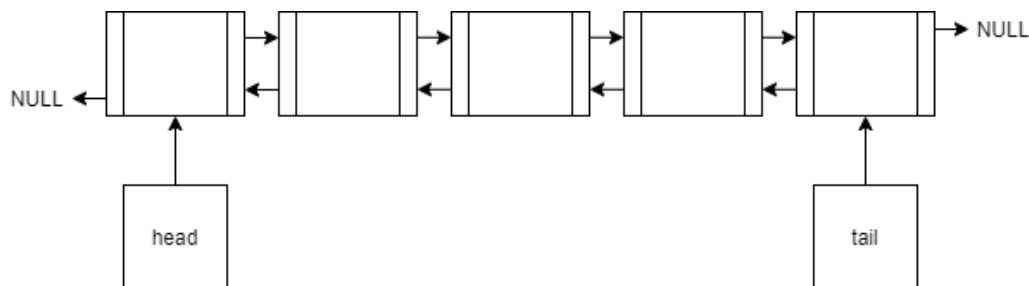


Рис. 3.2 – Схема структуры данных двусвязный список

3.2 TruckBrand

Структурой данных **перечисление** представлены марки грузовиков. Здесь содержатся все возможные значения марок грузовиков транспортной компании.

Схема представлена на рис. 3.3

Truck Brand
Volvo = 1, MercedesBenz = 2, Man = 3, Daf = 4, Scania = 5, Iveco = 6, Renault = 7, Nissan = 8, Tatra = 9, Mitsubishi = 10, NONE = 0

Рис. 3.3 – Перечисление TruckBrand

3.3 Destination

Возможные маршруты поставки содержатся в структуре данных **перечисление**. Здесь содержатся все возможные значения маршрутов транспортной компании.

Схема представлена на рис. 3.4

Destination
Omsk = 1, Kaliningrad = 2, StPetersburg = 3, Novgorod = 4, Sochi = 5, Murmansk = 6, NONE = 0

Рис. 3.4 – destination

3.4 Truck

Представление грузовика в памяти программы выполнено в виде структуры Truck со следующими полями:

- int id; – идентификатор грузовика
- float capacity; – максимальна масса груза
- int transportation_distance; – дальность перевозки
- TruckBrand brand; – марка
- Truck *prev; – указатель на предыдущий элемент
- Truck *next; – указател на следующий элемент

Схема на рис. 3.5.

Truck
int id; float capacity; int transportation_distance; Truck Brand brand; Truck *prev; Truck *next;

Рис. 3.5 – truck

3.5 TruckList

Для управления списком грузовиков используется структура TruckList со следующими полями:

- Truck *head; – указатель на первый элемент списка
- Truck *tail; – указатель на последний элемент списка
- int size; – количество элементов в списке

Структура так же содержит методы для управления списком. Схема на рис. 3.6.

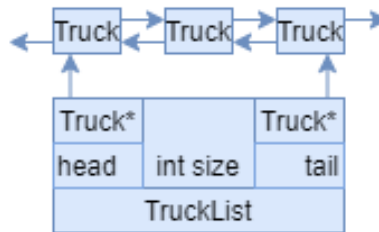


Рис. 3.6 – truck_list

3.6 TruckDataBase

Для управления базой данных грузовиков, её загрузки и обновления, используется структура TruckDataBase со следующими полями:

- TruckList list; – список
- std::string db_path; – путь к файлу с базой данных
- ScheduleDataBase *schedule; – указатель на расписание

Структура так же содержит методы для управления базой данных. Схема на рис. 3.7.

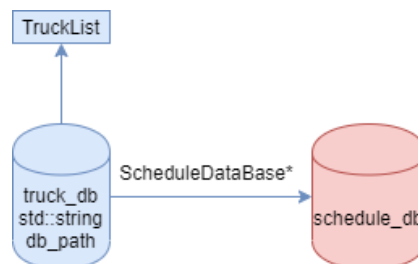


Рис. 3.7 – truck_db

3.7 Route

Представление маршрута в памяти программы выполнено в виде структуры Route со следующими полями:

- Destination destination; – точка прибытия
- int id; – идентификатор маршрута
- int distance; – дальность перевозки (в одну сторону)
- int loading_time; – время погрузки/разгрузки
- int drivers; – количество водителей
- int target_time_in_transit; – время в пути (в одну сторону)
- Route *prev; – указатель на предыдущий элемент
- Route *next; – указатель на следующий элемент

Схема на рис. 3.8.

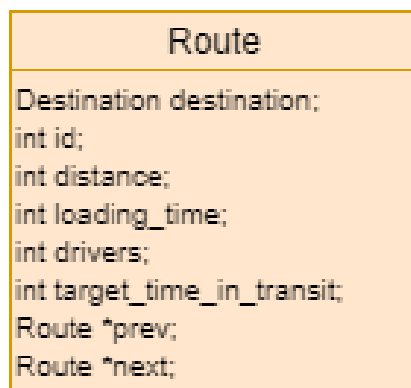


Рис. 3.8 – route

3.8 RouteList

Для управления списком маршрутов используется структура RouteList со следующими полями:

- Route *head; – указатель на первый элемент списка
- Route *tail; – указатель на последний элемент списка

- `int size;` – количество элементов в списке

Структура так же содержит методы для управления списком.
Схема на рис. 3.9.

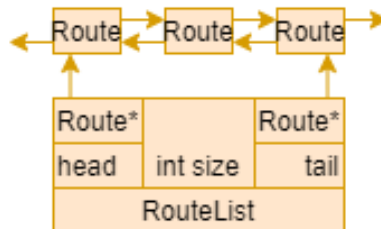


Рис. 3.9 – routelist

3.9 RouteDataBase

Для управления базой данных маршрутов, её загрузки и обновления, используется структура RouteDataBase со следующими полями:

- `RouteList list;` – список
- `std::string db_path;` – путь к файлу с базой данных

Структура так же содержит методы для управления базой данных.
Схема на рис. 3.10.

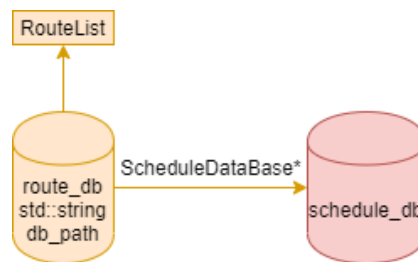


Рис. 3.10 – routedb

3.10 Driver

Представление водителя в памяти программы выполнено в виде структуры Driver со следующими полями:

- `int id;` – идентификатор
- `std::string surname;` – фамилия

- `std::string name;` – имя
- `std::string patronymic;` – отчество
- `TruckBrand truck_brand;` – марка грузовика
- `Driver *prev;` – указатель на предыдущий элемент
- `Driver *next;` – указатель на следующий элемент

Схема на рис. 3.11.

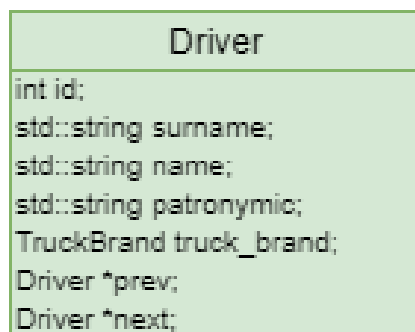


Рис. 3.11 – driver

3.11 DriverList

Для управления списком водителей используется структура `DriverList` со следующими полями:

- `Driver *head;` – указатель на первый элемент списка
- `Driver *tail;` – указатель на последний элемент списка
- `int size;` – количество элементов в списке

Структура так же содержит методы для управления списком.

Схема на рис. 3.12.

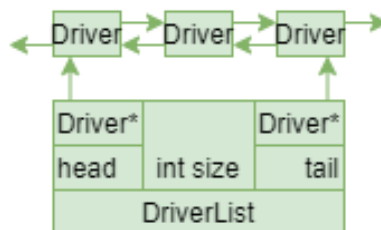


Рис. 3.12 – driverlist

3.12 DriverDataBase

Для управления базой данных водителей, её загрузки и обновления, используется структура DriverDataBase со следующими полями:

- DriverList list; – список
- std::string db_path; – путь к файлу с базой данных
- ScheduleDataBase *schedule; – указатель на расписание

Структура так же содержит методы для управления базой данных. Схема на рис. 3.13.

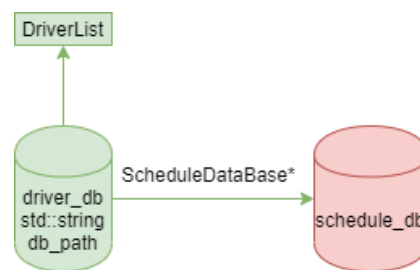


Рис. 3.13 – driverdb

3.13 Delivery

Представление поставки в памяти программы выполнено в виде структуры Delivery со следующими полями:

- std::time_t start; – время начала поставки
- std::time_t end; – время окончания поставки
- int *drivers_ids; – идентификаторы водителей
- int drivers; – количество водителей
- int truck_id; – идентификатор грузовика
- Delivery *next; – указатель на предыдущий элемент
- Delivery *prev; – указатель на следующий элемент

Схема на рис. 3.14.

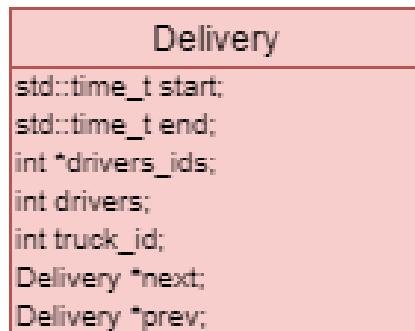


Рис. 3.14 – delivery

3.14 Schedule

Для управления списком поставок используется структура Schedule со следующими полями:

- Delivery *head; – указатель на первый элемент списка
- Delivery *tail; – указатель на последний элемент списка
- int size; – количество элементов в списке

Структура так же содержит методы для управления списком. Схема на рис. 3.15.

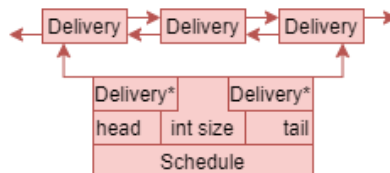


Рис. 3.15 – schedule

3.15 ScheduleDataBase

Для управления базой данных поставок, её загрузки и обновления, используется структура DriverDataBase со следующими полями:

- Schedule list; – список
- std::string db_path; – путь к файлу с базой данных

Структура так же содержит методы для управления базой данных. Схема на рис. 3.16.

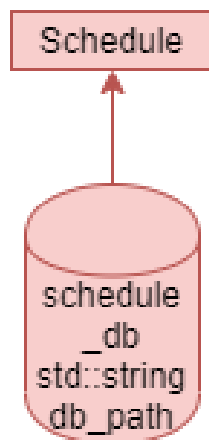


Рис. 3.16 – scheduledb

3.16 DataBases

Объединяет все базы данных структура DataBases со следующими полями:

- ScheduleDataBase schedule_db; – база данных с расписанием
- TruckDataBase truck_db; – база данных с грузовиками
- DriverDataBase driver_db; – база данных с водителями
- RouteDataBase route_db; – база данных с маршрутами

Схема на рис. 3.17.

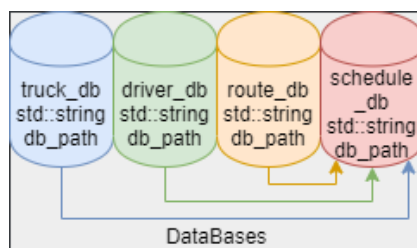


Рис. 3.17 – dbs

3.17 AdministratorConsole

Консоль администратора представлена структурой AdministratorConsole со следующими полями:

- `std::string password;` – пароль, вводимый пользователем
- `std::string real_password;` – заданный пароль (его хэш)
- `const char BACKSPACE=127;` – ascii код символа `backspace`
- `const char RETURN=10;` – ascii код символа `return`
- `DataBases *dbs;` – указатель на базы данных

Структура так же содержит методы для управления базами данных. Схема на рис. 3.18.

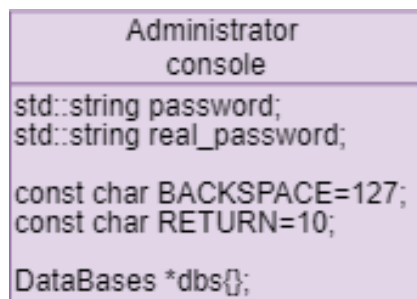


Рис. 3.18 – administratorconsole

3.18 Date

Для управления датой и временем используется структура `Date` со следующими полями:

- `std::string default_format;` – формат преобразования строки по умолчанию
- `std::tm *date;` – структура для хранения времени. необходима для получения форматированной строки со временем.
- `std::time_t time;` – время в секундах начиная с 1 янв 1970 г., необходимо для сравнительных и вычислительных действий в программе.

Схема на рис. 3.19.

Date
std::string default_format{};
std::tm *date{};
std::time_t time{};

Рис. 3.19 – date

3.19 Request

Запрос на поставку представлен в виде структуры Request со следующими полями:

- float cargo_weight; – масса груза
- Date departure_date; – дата отправки
- Date arrival_date; – дата прибытия (заполняется программой)
- Route *target_route; – маршрут (заполняется программой)
- Destination destination; – точка прибытия
- TruckBrand truck_brand; – марка грузовика

Схема на рис. 3.20.

Request
float cargo_weight{};
Date departure_date{};
Date arrival_date{};
Route *target_route{};
Destination destination;
TruckBrand truck_brand;

Рис. 3.20 – request

3.20 RequestHandler

Для обработки запроса на поставку используется структура RequestHandler со следующими полями:

- Request request; – запрос
- std::string request_path; – путь к файлу с запросом
- DataBases *dbs; – указатель на базы данных

Схема на рис. 3.21.



Рис. 3.21 – requesthandler

4 Описание пользовательских функций

Все функции и модули, реализованные в ходе выполнения данной курсовой работы, представлены в таблице 1.

Таблица 1 – Описание пользовательских функций и модулей программы

Имя модуля	Имя функции	Назначение	Параметры функции	Возвращаемое значение функции
truck_brand.h	str	Вывод марки грузовика в виде строки	TruckBrand truck_brand	std::string

Программа разделена на два интерфейса: пользовательский, для получения и обработки запросов, и администраторский, для редактирования и просмотра баз данных.

Алгоритм программы

Тестирование программы

Выводы

Вывод

Список литературы

- [1] ЧТО ТАКОЕ «НАЛОГ НА ПРОФЕССИОНАЛЬНЫЙ ДОХОД» // nalog.ru URL: <https://npd.nalog.ru/> (дата обращения: 28.11.2020 г.)
- [2] Список // Викиконспекты URL: <https://neerc.ifmo.ru/wiki/index.php?title=Список> (дата обращения: 07.12.2020 г.)

Приложение А. Листинг программного ко- да