

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное учреждение высшего образования
«Петрозаводский государственный университет»
Физико-технический институт
Кафедра информационно-измерительных систем и физической электроники

ТРАНСПОРТНАЯ КОМПАНИЯ

отчет по лабораторной работе

Автор работы:
студент группы 21316
_____ В. Д. Клевцов
«26» декабря 2022 г.

Научный руководитель:
канд. физ.-мат. наук, доцент
_____ А. В. Бульба
«26» декабря 2022 г.

Петрозаводск
2022 год

Содержание

Цель.....	3
Кратко о программной реализации.....	3
Ход разработки	3
1) Краткое словесное описание сюжета	3
2) Описание имеющихся у заказчика материалов	4
3) Actors	4
4) Диаграмма вариантов использования.....	4
5) Описание каждого варианта использования.....	5
6) Диаграммы действий.....	6
7) Предварительный список существительных	6
8) Уточненный список существительных	7
9) Список атрибутов	7
10) Список сообщений.....	8
11) Диаграмма классов	8
12) Диаграммы последовательности	8
13) Листинг заголовочных файлов	10
14) Листинг исходных файлов	12
15) Руководство пользователя	15
Заключение.....	16

Цель: Рассмотреть простой пример выполнения заказа на разработку программного обеспечения и, в качестве самостоятельного задания, выполнить аналогичный проект (в контексте транспортной компании).

Кратко о программной реализации:

Для реализации поставленной задачи была использована среда программирования Qt Creator 5.4.2, в качестве языка программирования был выбран C++. В процессе работы над проектом созданы заголовочные файлы:

1. Drivers.h
2. DriversInputScreen.h
3. DriversList.h
4. tc.h

Ход разработки:

1) Краткое словесное описание сюжета:

Сфера транспортных услуг включает в себя не только большие компании, но и маленькие, которые занимаются перевозками по городу или за его пределами. Такие компании обычно обладают небольшим количеством сотрудников, программа TransportCompany поможет снизить нагрузку с персонала.

Функционал программы позволяет:

- вводить имя водителя, а также номер рейса, который присваивается водителю
- записывать ежемесячные доходы, которые приносит каждый водитель компании
- учитывать расходы, а именно дату, категорию, получателя и сумму

Исходя из данных, которые пользователь занесет в ходе работы, программа сможет вывести:

- список всех водителей компании с их номером рейса, фамилией и именем
- таблицу с доходами водителей по месяцам
- расходы водителей за весь период, с описанием каждого из них
- годовой отчет, который включает в себя все доходы и убытки в течение года.

Также, пользователь может вывести отчет не только за год, но и за любой промежуток времени, начиная с начала года

Нельзя не отметить тот факт, что при подготовке годового отчета шанс допустить ошибку снижается, так как все подсчеты ведутся программой.

2) Описание имеющихся у заказчика материалов:

Таблица 1. Список водителей.

Номер рейса	Имя водителя
101	Виктор Хмурый
402	Иван Кузнецов
576	Кирилл Золов
368	Егор Лоскутов
204	Илья Мясников

Таблица 2. Доход от водителей.

Номер рейса	Янв	Фев	Март	Апр	Май	Июнь	Июль	Авг	Сен	Окт	Нояб	Дек
101	0	0	2000	2250	0	0	0	0	0	0	0	0
402	1200	2600	600	2200	2100	0	2420	1630	0	0	0	0
576	0	0	0	0	0	0	0	0	0	0	1200	2150
368	0	0	0	0	0	0	0	0	0	600	1600	1840
204	0	0	0	0	0	0	0	0	0	400	890	1360

Таблица 3. Годовой отчет

Доходы	45000
Расходы	14000
Итого	31000

3) Actors:

В качестве действующего лица будет выступать диспетчер, он будет добавлять водителей, следить за их прибылью в течение месяца и получать годовой отчет.

4) Диаграмма вариантов использования:



5) Описание каждого варианта использования:

Запуск программы:

Когда запускается программа, на экран должно выводиться меню, из которого пользователь.

может выбрать нужное действие. Это может называться экраном пользовательского интерфейса.

Добавить водителя:

На экране должно отобразиться сообщение, в котором программа просит пользователя ввести имя водителя и номер рейса. Эта информация должна заноситься в таблицу.

Ввести данные о рейсе:

Экран ввода данных о рейсе содержит сообщение, из которого пользователь узнает, что ему необходимо ввести имя водителя, месяц рейса, а также полученную сумму денег.

Программа просматривает список водителей и по номеру рейса находит соответствующую запись в таблице доходов от рейса. Если водитель впервые выходит в рейс, в этой таблице создается новая строка и указанная сумма заносится в столбец того месяца, а котором производится рейс. В противном случае значение вносится в существующую строку.

Ввести расходы:

Экран ввода расхода должен содержать приглашение пользователю на ввод имени получателя (или названия организации), суммы оплаты, дня и месяца, в который производится оплата, категории расходов. Затем программа создает новую строку, содержащую эту информацию, и вставляет ее в таблицу расходов.

Вывести список водителей компании:

Программа выводит на экран список водителей, каждая строка списка состоит из двух полей: номер рейса и имя водителя.

Вывести таблицу доходов:

Каждая строка таблицы, которую выводит программа, состоит из номера рейса и значения оплаты рейса за месяц.

Вывести таблицу расходов:

Каждая строка таблицы, которую выводит программа, состоит из номера рейса и значения ежемесячной оплаты рейса.

Вывести годовой отчет:

Программа выводит годовой отчет, состоящий из:

суммарного дохода рейсов за прошедший год;

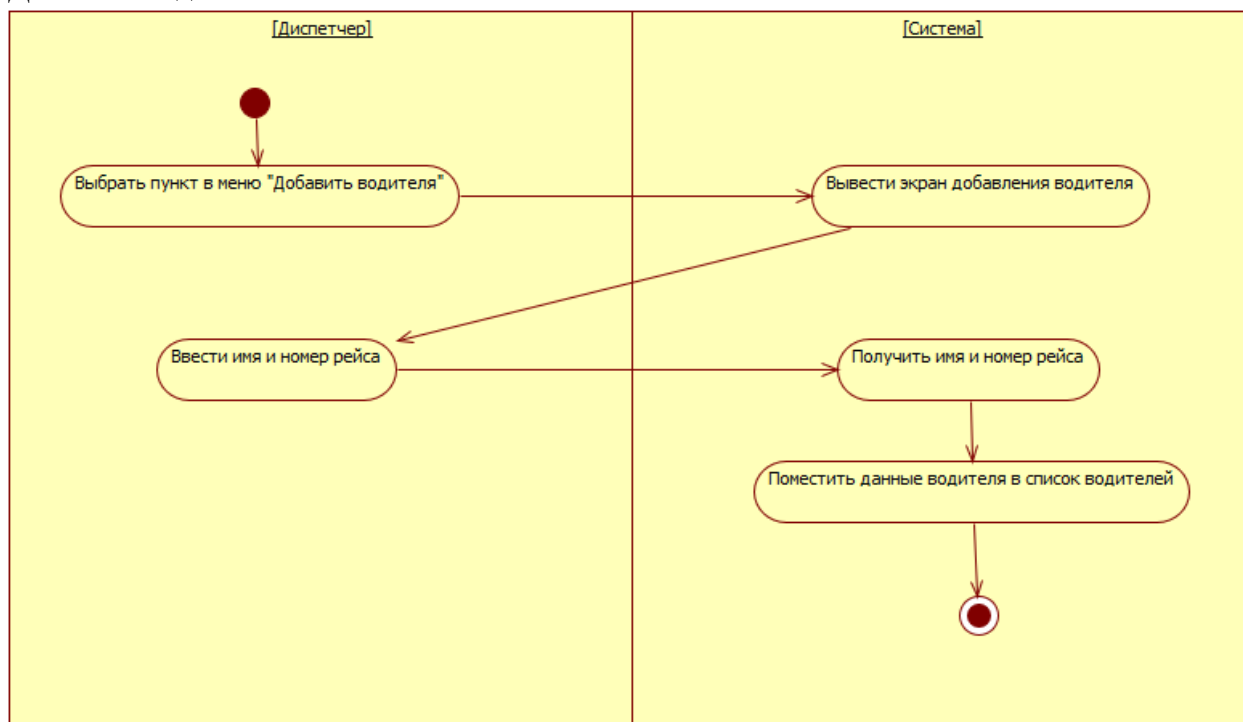
списка всех расходов с указанием категории;

общая сумма расходов;

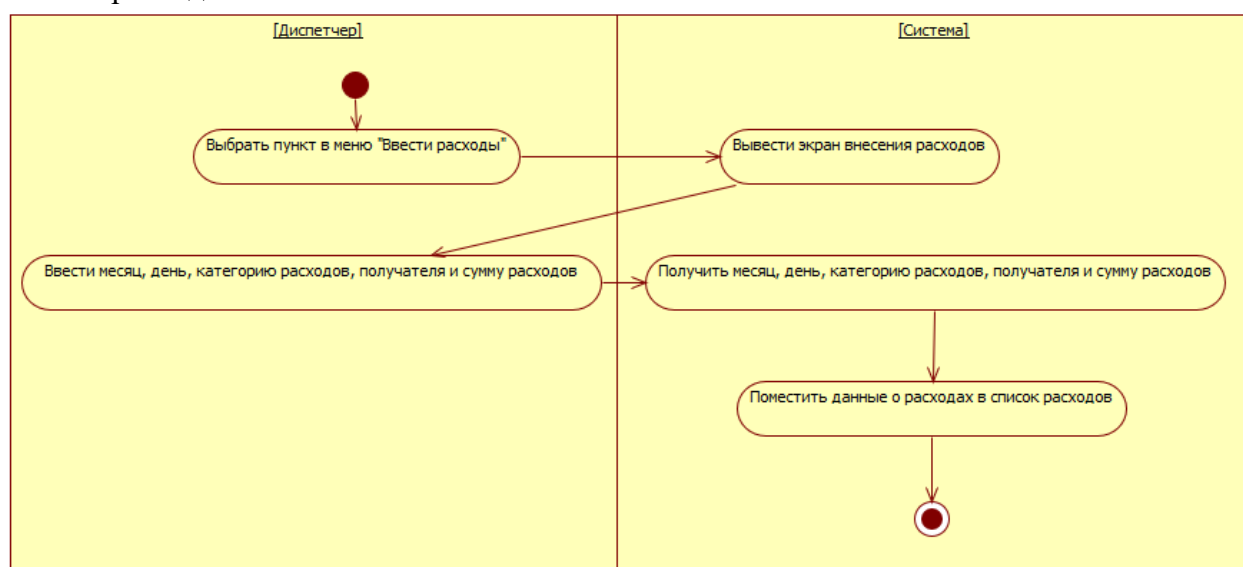
результатирующего годового баланса (доходы/убытки)

6) Диаграммы действий:

Добавить водителя:



Ввести расходы:



7) Предварительный список существительных:

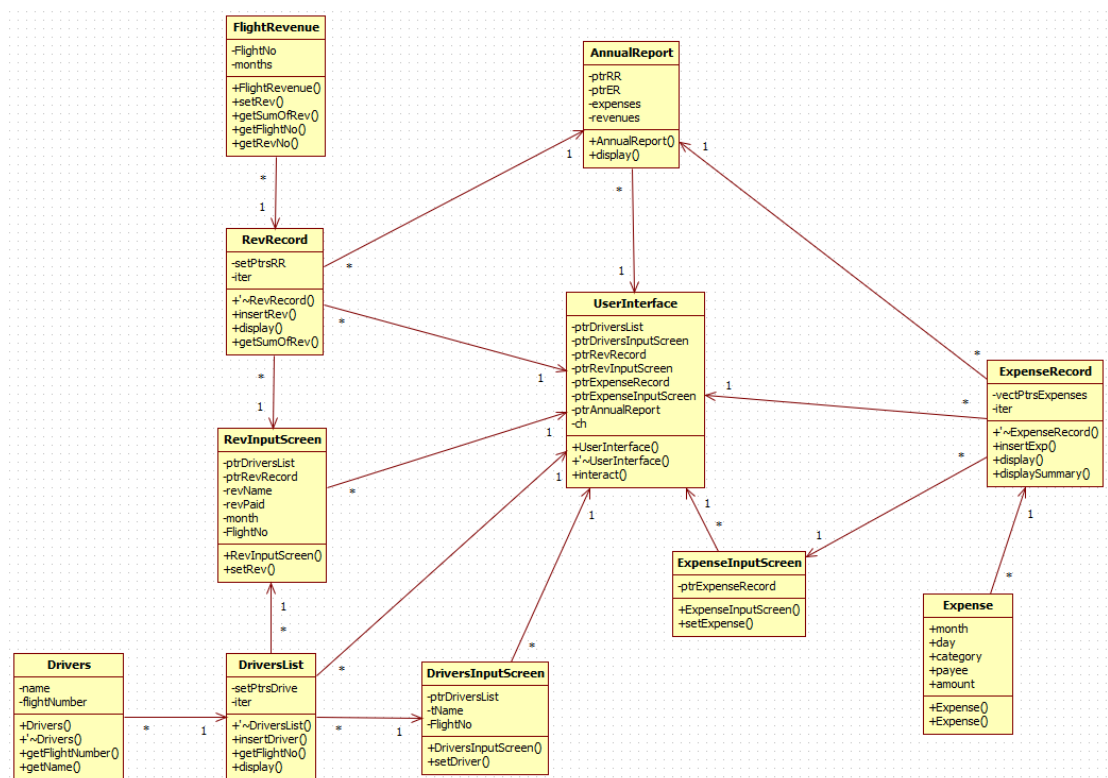
1. Экран интерфейса пользователя.
2. Водитель
3. Экран ввода данных о рейсе
4. Имя водителя
5. Номер рейса
6. Список водителей
7. Данные о рейсе
8. Экран ввода данных о рейсе
9. Месяц
10. Полученная сумма денег
11. Таблица доходов

12. Строка оплаты рейса
 13. Расход
 14. Экран ввода расходов
 15. Получатель
 16. Сумма оплаты
 17. День
 18. Категория расходов
 19. Строка таблицы расходов
 20. Таблица расходов
 21. Годовой отчет
 22. Суммарные доходы рейсов
 23. Суммарные расходы с указанием категории
 24. Баланс
- 8) Уточненный список существительных:
1. Экран интерфейса пользователя.
 2. Водитель
 3. Экран ввода данных о рейсе
 4. Список водителей
 5. Экран ввода данных о рейсе
 6. Таблица доходов
 7. Строка оплаты рейса
 8. Расход
 9. Экран ввода расходов
 10. Таблица расходов
 11. Годовой отчет
- 9) Список атрибутов:
1. `UserInterface`
 - a) `ptrDriversList` - указатель на класс список водителей
 - b) `ptrDriversInputScreen` - указатель на класс экран ввода водителей
 - c) `ptrRevRecord` - указатель на класс записей дохода
 - d) `ptrRevInputScreen` - указатель на класс экран добавления дохода
 - e) `ptrExpenseRecord` - указатель на класс записей о затратах
 - f) `ptrExpenseInputScreen` - указатель на класс ввода расходов
 - g) `ptrAnnualReport` - указатель на класс годового отчета
 - h) `ch` – переменная для работы switch
 2. `DriversList`
 - a) `setPtrsDrive` – указатель на список класса водителей
 - b) `iter` – итератор для работы со списком
 3. `DriversInputScreen`
 - a) `ptrDriversList` – указатель на список водителей
 - b) `tName` – имя водителя
 - c) `FlightNo` – номер рейса
 4. `Drivers`
 - a) `name` – имя водителя
 - b) `flightNumber` – номер рейса

10) Список сообщений:

1. **UserInterface**
 - a) `UserInterface()` – конструктор
 - b) `~UserInterface()` - деструктор
 - c) `interact()` – функция для взаимодействия пользователя с системой
2. **DriversList**
 - a) `~DriversList()` - деструктор
 - b) `insertDriver()` – вставляет нового водителя в конец списка водителей
 - c) `getFlightNo()` – возвращает номер рейса водителя по имени водителя
 - d) `display()` - вывод списка водителей
3. **DriversInputScreen**
 - a) `DriversInputScreen()` – конструктор
 - b) `setDriver()` – добавляет данные о водителе и заносит в список водителей
4. **Drivers**
 - a) `Drivers()` – конструктор
 - b) `~Drivers()` - деструктор
 - c) `getFlightNumber()` – возвращает номер рейса
 - d) `getName()` – возвращает имя водителя

11) Диаграмма классов:



12) Диаграммы последовательности:

Диаграмма последовательностей “Добавить водителя”:

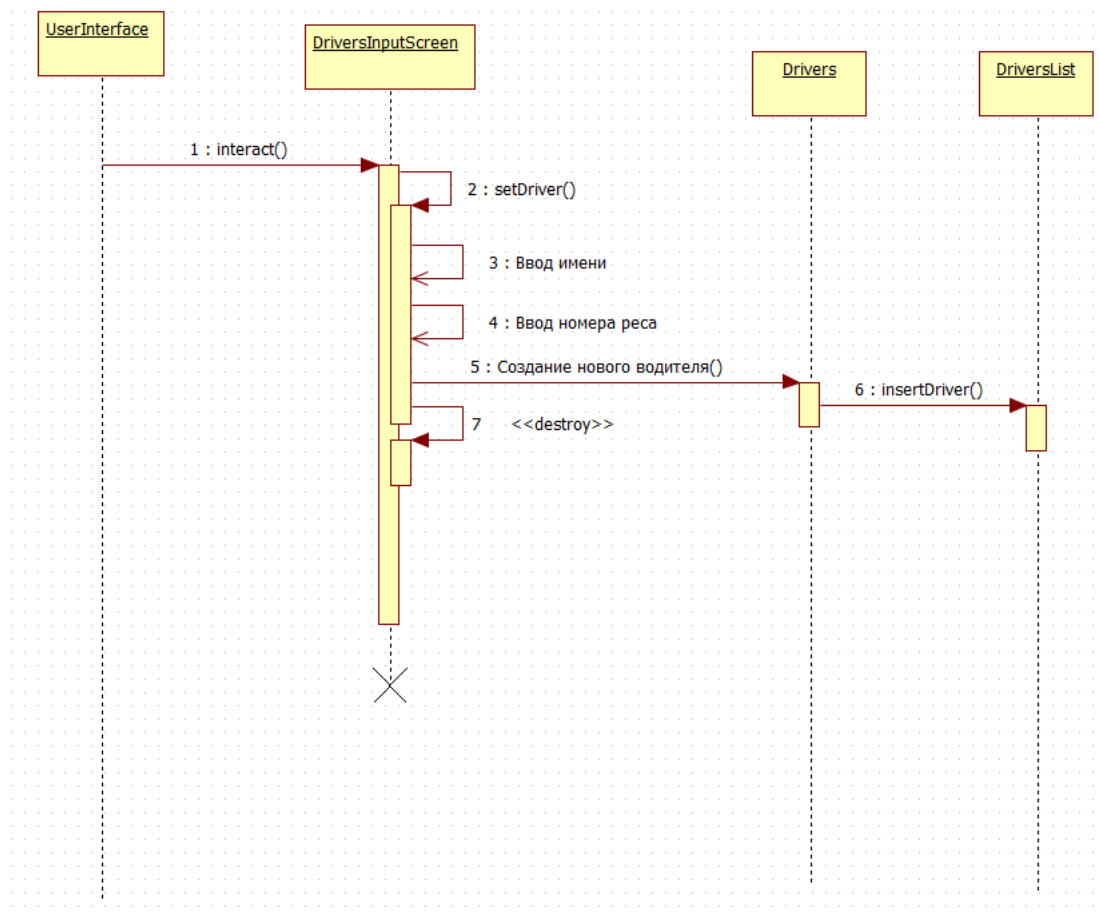


Диаграмма последовательностей “Вывести список водителей компании”:

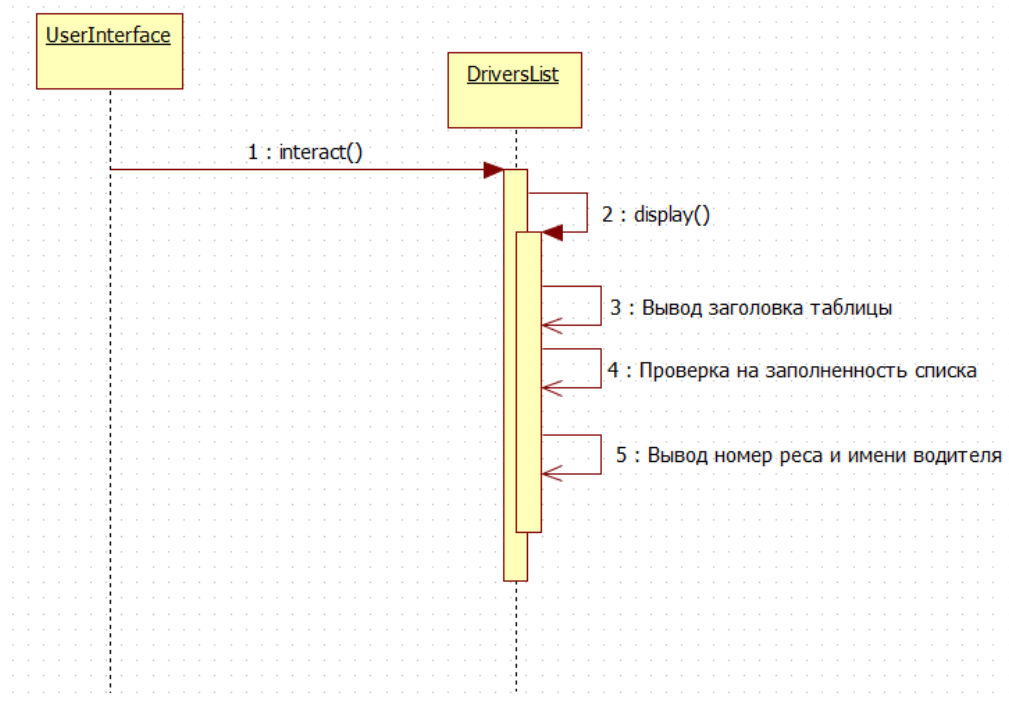
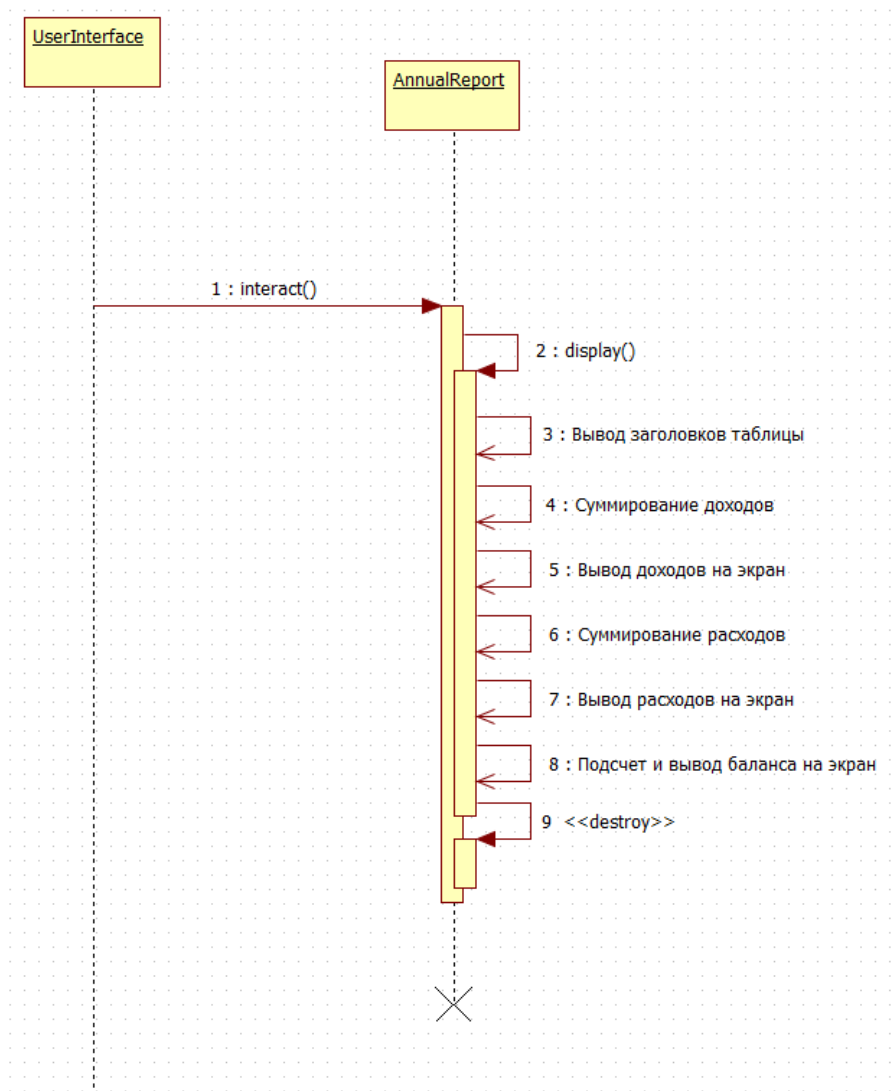


Диаграмма последовательностей “Вывести годовой отчет”:



13) Листинг заголовочных файлов:

tc.h

```

#include <list>
#include "Drivers.h"
#include "DriversList.h"
#include "DriversInputScreen.h"
using namespace std;

// глобальные методы //
void getaLine(string& inStr); // получение строки текста
char getaChar(); // получение символа

//Класс UserInterface//
//Главный класс для запуска приложения:
//этот класс определяет взаимодействие пользователя с программой
class UserInterface
{
private:
    DriversList* ptrDriversList;
    DriversInputScreen* ptrDriversInputScreen;
    char ch;
public:
    UserInterface();
    ~UserInterface();
    void interact();
};
  
```

```
#endif // TC_H
```

DriversList.h

```
//DriversList.h
#ifndef DRIVERSLIST
#define DRIVERSLIST
#include "Drivers.h"
using namespace std;
//класс DriversList — список всех водителей.
//Он содержит множество указателей на класс Drivers
// и оперирует ими при выводе
class DriversList
{
private:
    // установить указатели на водителей
    list <Drivers*> setPtrsDrive; // указатели на класс водителей
    list <Drivers*>::iterator iter; //итератор
public:
    ~DriversList(); // деструктор (удаление водителей)
    void insertDriver(Drivers*); // добавить водителя в список
    int getFlightNo(string); // возвращает номер рейса
    void display(); // вывод списка водителей
};
#endif // DRIVERSLIST
```

DriversInputScreen.h

```
//DriversInputScreen.h
#ifndef DRIVERSINPUTSCREEN
#define DRIVERSINPUTSCREEN
#include "UserInterface.h"
//класс DriversInputScreen. Это класс, отвечающий за отображение «экрана»,
//куда пользователь может ввести данные о новом водителе:
class DriversInputScreen
{
private:
    DriversList* ptrDriversList;
    string tName;
    int FlightNo;
public:
    DriversInputScreen(DriversList* ptrDL) : ptrDriversList(ptrDL)
    {
        /* тут пусто */
    }
    void setDriver(); // добавить данные о водителе
};
#endif // DRIVERSINPUTSCREEN
```

Drivers.h

```
//Drivers.h
#ifndef DRIVERS
#define DRIVERS
#include "UserInterface.h"
using namespace std;
// класс Drivers (водители) //
```

```

//Он хранит имя водителя и номер рейса, в котором он находится.
class Drivers
{
private:
    string name; // имя водителя
    int flightNumber; // номер рейса, в котором он находится
public:
    Drivers(string n, int aNo);
    ~Drivers();
    int getFlightNumber(); //возвращает номер рейса водителя
    string getName(); //возвращает имя водителя
};
#endif // DRIVERS

```

14) Листинг исходных файлов:

tc.cpp

```

//tc.cpp
#include <iostream>
#include "tc.h"
void getaLine(string & inStr) // получение строки текста
{
    char temp[21];
    cin.get(temp, 20, '\n');
    cin.ignore(20, '\n'); //число пропускаемых символов и символ разделения
    inStr = temp;
}
//-----
char getaChar() // получение символа
{
    char ch = cin.get();
    cin.ignore(80, '\n'); //число пропускаемых символов и символ разделения
    return ch;
}
//методы класса UserInterface//
UserInterface::UserInterface()
{
    ptrDriversList = new DriversList;
}
UserInterface::~~UserInterface()
{
    delete ptrDriversList;
}
void UserInterface::interact()
{
    while (true)
    {
        cout << "\n Click to add a driver 'a', \n" //Нажмите для
добавления водителя
        << " Click to withdraw drivers 'd', \n" //Нажмите
для вывода водителей
        << " Click to exit the program 'q'\n"; //Нажмите для
выхода из программы
        ch = getaChar();
        switch (ch)
        {
            case 'a': ptrDriversInputScreen = new
DriversInputScreen(ptrDriversList);
ptrDriversInputScreen->setDriver();
delete ptrDriversInputScreen;

```

```

        break;
    case 'd': ptrDriversList->display();
        break;
    case 'q':
        cout << "The program is completed"; // Программа завершена
        return;
        break;
    default: cout << "Unknown output function\n"; //Неизвестная
функция вывода
        break;
    } // конец switch
} // конец while
} // конец interact()

```

TransportCompany.cpp

```

//TransportCompany.cpp
#include "tc.h"
int main()
{
    UserInterface theUserInterface;
    theUserInterface.interact();
    return 0;
}

```

DriversList.cpp

```

//DriversList.cpp
#include "UserInterface.h"

DriversList::~DriversList() // деструктор
{
    while (!setPtrsDrive.empty()) // удаление всех водителей,
    { // удаление указателей из контейнера
        iter = setPtrsDrive.begin();
        delete* iter;
        setPtrsDrive.erase(iter);
    }
}

void DriversList::insertDriver(Drivers* ptrD)
{
    setPtrsDrive.push_back(ptrD); // вставка нового водителя в список
}

int DriversList::getFlightNo(string tName) // получить номер рейса по имени водителя
{
    int FlightNo;
    iter = setPtrsDrive.begin();
    while (iter != setPtrsDrive.end())
    { // поиск водителя в списке (достаем у каждого водителя номер рейса)
        FlightNo = (*iter)->getFlightNumber();
        if (tName == ((*iter)->getName())) // сравниваем по именам и
        {
            // если получившаяся пара совпадает - значит,
            //мы нашли запись об этом водителе в списке, в этом случае
            return FlightNo; // возвращаем номер его рейса
        }
    }
}

```

```

        iter++;
    }
    return -1; // если нет - возвращаем -1
}

void DriversList::display() // вывод списка водителей
{
    cout << "\nNumber#\tName Driver\n-----\n";
    if (setPtrsDrive.empty()) // если список водителей пуст
        cout << "***No driver***\n" << endl; // выводим запись, что он пуст)
    else
    {
        iter = setPtrsDrive.begin();
        while (iter != setPtrsDrive.end()) // распечатываем всех водителей
        {
            cout << (*iter)->getFlightNumber() << "  || " << (*iter)->getName() << endl;
            *iter++;
        }
    }
}

```

DriversInputScreen.cpp

```

//DriversInputScreen.cpp
#include "UserInterface.h"

void DriversInputScreen::setDriver() // добавить данные о водителе
{
    cout << "Enter the driver's name (Alexander Masloy): " << endl; // Введите имя водителя
    (Александр Маслов):
    getaLine(tName);
    cout << "Enter the flight number (515): " << endl; // Введите номер рейса (515):
    cin >> FlightNo;
    cin.ignore(80, '\n');
    Drivers* ptrDriver = new Drivers(tName, FlightNo); // создать водителя
    ptrDriversList->insertDriver(ptrDriver); // занести в список водителей
}

```

Drivers.cpp

```

//Drivers.cpp
#include "UserInterface.h"

//в конструкторе задаём имя водителя и номер рейса
Drivers::Drivers(string n, int aNo) : name(n), flightNumber(aNo)
{
    /* тут пусто */
}

Drivers::~~Drivers() // деструктор
{
    /* тут тоже пусто */
}

int Drivers::getFlightNumber() //getter возвращает номер рейса водителя
{

```

```

    return flightNumber;
}

string Drivers::getName() //геттер возвращает имя водителя
{
    return name;
}

```

15) Руководство пользователя:

При запуске программы выводится меню:

```

Click to add a driver 'a',
Click to record the driver's income 'b',
Click to record expenses 'c',
Click to withdraw drivers 'd',
Click to withdraw revenue 'e',
Click to display expenses 'f',
Click to display the annual report 'g',
Click to exit the program 'q'.

```

1. При выборе 'a' Добавить водителя, пользователю будет предложено ввести имя водителя и его номер рейса, после чего выведется меню.

```

a
Enter the driver's name (Alexander Masloy):
Alex Berger
Enter the flight number (515):
316

```

2. При выборе 'b' Заполнить доходы водителей, пользователю будет предложено ввести имя водителя, сумму, которую он принес компании, и месяц, в котором был доход. После выведется меню.

```

b
Enter the driver's name: Alex Berger
Enter the amount of income (800.50):
5000
Enter the payment month number (1-12):
1

```

3. При выборе 'c' Заполнить расходы, пользователю будет предложено ввести месяц, день и категорию расходов, получателя, а также потраченную сумму. После выведется меню.

```

c
Enter the month (1-12): 2
Enter the day (1-31): 22
Enter the expense category (fuel, toll roads, repair, taxes): fuel
Enter the recipient (Gazprom, rosavtodor, service station, Russia): Lukoil
Enter the amount (350.65): 2100

```

4. При выборе 'd' Вывести водителей, пользователю будет выведена таблица: номер рейса || имя водителя. После выведется меню.

```

d
Number# Name Driver
-----
316 || Alex Berger
101 || Viktor Maslov
220 || Maxim Nak

```

5. При выборе 'е' Вывести доходы, пользователю будет выведена таблица с ежемесячным доходом от водителей, которые учитываются по номеру рейса водителя. После выведется меню.

Number	Jan	Feb	Mar	Apr	May	June	July	Aug	Sept	Oct	Nov	Dec
316	5000	0	0	0	0	0	0	0	0	0	0	0
220	0	3500	0	4000	0	0	0	0	0	0	0	0

6. При выборе 'f' Вывести расходы, пользователю будет выведена таблица расходов с указанием даты, получателя, суммы и категории расходов. После выведется меню.

Date	Recipient	Amount	Category
2/22	Lukoil	2100	fuel

7. При выборе 'g' Вывести годовой отчет, пользователю будет выведена таблица, где будет учтена вся прибыль за прошедший период, указаны расходы, с указанием категории и цены, а также подведен итог, заработала ли компания за этот промежуток времени или же понесла убытки. После выведется меню.

Annual Report			
Income			
	Payment for the flight:	700000	
Expenses			
	fuel	30000	
	toll roads	5000	
Total expenses:		35000	
Balance:		665000	

8. При выборе 'q' Выход из программы, программа завершается, пользователя уведомят об этом.

q
The program is completed

История проекта на GitHub



Author	Commit Message	Date
Matvey Shevchov <shem1604@yandex.ru>	Added AnnualRecord.h and AnnualRecord.cpp. Renamed tc as UserInterface	2022-12-26 20:17:59
Matvey Shevchov <shem1604@yandex.ru>	Added Expense.h, ExpenseRecord.h, ExpenseInputScreen.h, Expense.cpp, ExpenseRecord.cpp and ExpenseInputScreen.cpp	2022-12-26 19:53:36
Matvey Shevchov <shem1604@yandex.ru>	Merge branch 'develop'	2022-12-26 19:30:42
Vladislav Klevtsov <klevtsov1204@mail.ru>	Add RevRecord.h, RevRecord.cpp, RevInputScreen.h and RevInputScreen.cpp	2022-12-24 19:24:47
Dusosed <dasorokin123@gmail.com>	Add FlightRevenue.cpp and FlightRevenue.h	2022-12-24 18:03:06
Dusosed <dasorokin123@gmail.com>	Add RevRecord, RevInputScreen and FlightRevenue classes	2022-12-24 18:43:44
Vladislav Klevtsov <klevtsov1204@mail.ru>	Add DriversInputScreen.h and DriversInputScreen.cpp	2022-12-23 23:03:43
Vladislav Klevtsov <klevtsov1204@mail.ru>	Add Drivers.h, Drivers.cpp, DriversList.h and DriversList.cpp	2022-12-23 22:56:23
Vladislav Klevtsov <klevtsov1204@mail.ru>	Program version 1. Added Drivers, DriversList and DriversInputScreen classes	2022-12-23 22:35:31

Адрес проекта: <https://github.com/he4to-hoboe/TransportCompany.git>

Заключение:

Использовалась среда разработки Qt Creator 5.4.2, применялась система контроля версий, все прецеденты реализованы, сбои и зависания не наблюдаются, реализована очистка памяти, использованы принципы раздельной компиляции, приложены диаграммы: прецедентов, действий, классов, последовательностей.