Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и анализа данных

Центр программной инженерии

ОТЧЕТ

к лабораторной работе №4 по дисциплине:

|  |
| --- |
| «Объектно-ориентированное программирование» |
| Разработка классов с использованием механизмов  наследования, полиморфизма и инкапсуляции. |

наименование темы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы: | ИСТб-19-2 |  |  |  | Комогорцева Ю.В. |
|  | шифр группы |  | подпись |  | Фамилия ИО |
| Проверил: | доцент |  |  |  | Маланова Т.В. |
|  | должность |  | подпись |  | Фамилия ИО |

Иркутск. 2020 г.

Оглавление

[Оглавление 2](#_Toc56728292)

[1 Постановка задачи 3](#_Toc56728293)

[2 Описание структуры пользовательского меню 4](#_Toc56728294)

[3 Проектирование классов 5](#_Toc56728295)

[4 Описание методов классов и спецификация локальных переменных 6](#_Toc56728296)

[5 Таблица тестов 10](#_Toc56728297)

[6 Результаты тестирования 16](#_Toc56728298)

[7 Код программы 20](#_Toc56728299)

[Список использованных источников 29](#_Toc56728300)

1 Постановка задачи

В соответствии с индивидуальным заданием описать иерархию

классов, для каждого класса описать поля и соответствующие методы

доступа к ним.

В зависимости от задания некоторые из этих классов (как минимум

один) являются абстрактными классами и служат для выделения общих

данных и поведения для других классов. Абстрактный класс должен

содержать как минимум один абстрактный метод, реализация которого у его

наследников должна различаться.

Помимо этого в общую часть задания входит разработка класса,

группирующего объекты описанных в соответствии с заданием классов.

Для выполнения задания необходимо создать некоторое количество

объектов, добавить их в группу используя предусмотренные методы класса

«группы» и для каждого из них вызвать унаследованный метод.

Индивидуальное задание

Реализовать следующие классы:

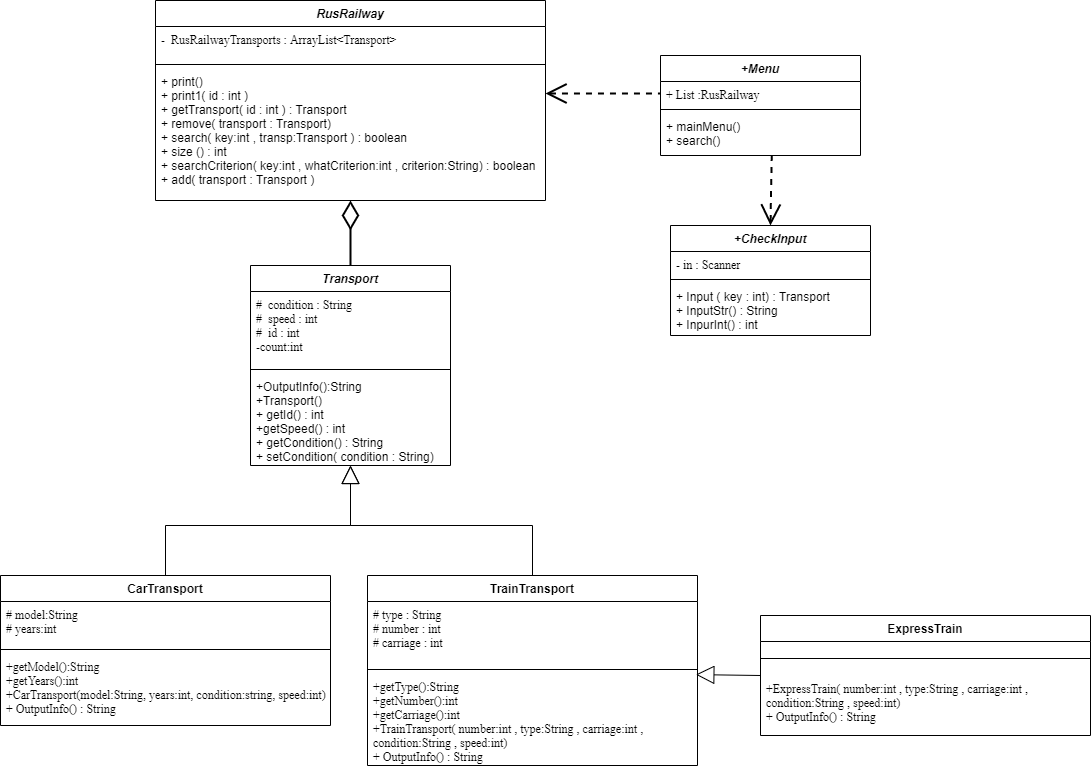
*Вариант 11*-автомобиль, поезд, транспортное средство, экспресс;

Разработать класс-группу(РЖД). Описать иерархию данных классов.

2 Проектирование классов

На рисунке 1 представлена UML диаграмма классов.

Рисунок 1 – UML диаграмма классов



Описание классов.

Menu-Класс реализующий общение с пользователем(меню)

CheckInput-Проверка вводимых значений

Transport-транспорт, родительский класс для CarTransport, TrainTransport, ExpressTrain

CarTransport-машины, наследник класса Transport

TrainTransport-поезд, наследник класса Transport

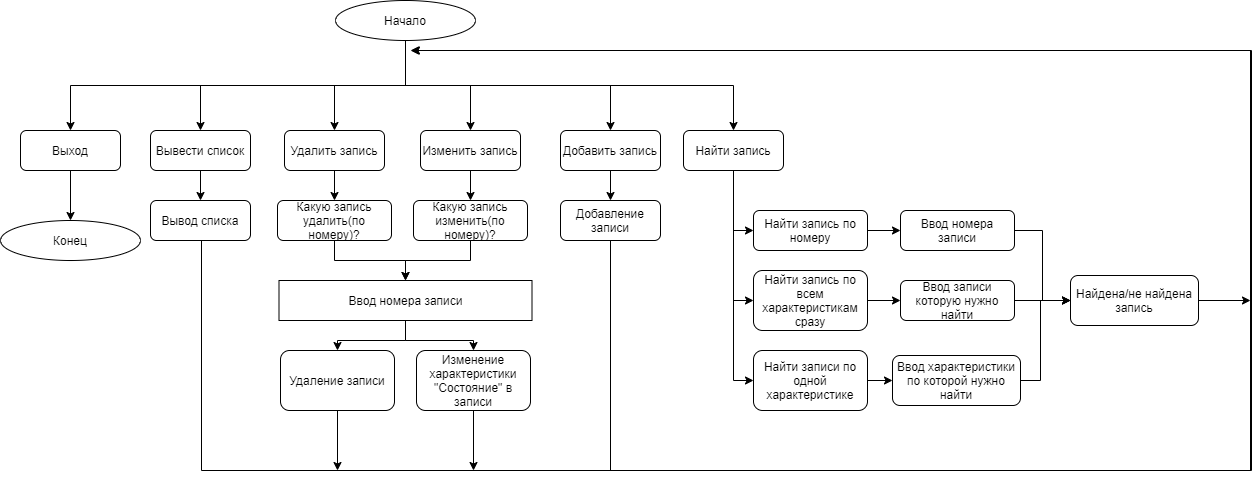
ExpressTrain-экспресс-поезд, наследник класса Transport и TrainTransport

RusRailway-– класс-группа для объектов класса Tranport

**3 Структура меню**

На рисунке 2 представлена структура пользовательского меню.

Рисунок 2 – Структура пользовательского меню



Классы, реализующие пользовательское меню:

Класс Menu (главное меню) – главный класс пользовательского меню,

предоставляющий интерфейс взаимодействия с пользователем.

Класс CheckInput – класс подменю, отвечающий за проверку ввода.

4 Описание методов классов и спецификация локальных переменных

Описание класса Transport

Таблица 1 – Поля абстрактного класса Transport

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | condition | Состояние транспорта | String | protected |
| 2 | speed | Скорость | int | protected |
| 3 | id |  | int | protected |
| 4 | count |  | int | private |

Таблица 2 – Конструкторы абстрактного класса State

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | Transport() | Конструктор по умолчанию |  | public |

Таблица 3 – Методы класса

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа | Абстрактный/  неабстрактный | Параметры |
| 1 | OutputInfo() | Для вывода информации об обьекте | String | public | Абстрактный |  |
| 2 | getId() | Геттер | int | private | Неабстрактный |  |
| 3 | getSpeed() | Геттер скорости | int | public | Неабстрактный |  |
| 4 | getCondition() | Геттер состояния | String | public | Неабстрактный |  |
| 5 | setCondition(String condition) | Сеттер состояния | void | public | Неабстрактный |  |

Описание класса CarTransport

Таблица 1 – Поля класса CarTransport

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | years | Год выпуска авто | int | protected |
| 2 | model | Модель | String | protected |

Таблица 2 – Конструкторы класса CarTransport

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | CarTransport |  |  | public |

Таблица 3 – Методы класса CarTransport

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа | Абстрактный/  неабстрактный | Параметры |
| 1 | getModel | Геттер модели | String | public | неабстрактный |  |
| 2 | getYears | Геттер года выпуска | int | public | неабстрактный |  |
| 3 | OutputInfo() | Метод для вывода информации об обьекте | String | public | неабстрактный |  |

Описание класса TrainTransport

Таблица 1 – Поля класса TrainTransport

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | type | Тип поезда: пассажирский/грузовой | String | protected |
| 2 | carriage | Количество вагонов | int | protected |
| 3 | number | Номер поезда | int | protected |

Таблица 2 – Конструкторы класса TrainTransport

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | TrainTransport | Конструктор по умолчанию |  | public |

Таблица 3 – Методы класса TrainTransport

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа | Абстрактный/  неабстрактный | Параметры |
| 1 | getType | Геттер типа поезда | String | public | неабстрактный |  |
| 2 | getNumber | Геттер номера поезда | int | public | неабстрактный |  |
| 3 | getCarriage | Геттер количества вагонов | int | public | неабстрактный |  |
| 4 | OutputInfo() | Метод для вывода информации об обьекте | String | public | неабстрактный |  |

**Описание класса ExpressTrain**

Таблица 1 – Поля класса ExpressTrain

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 |  |  |  |  |

Таблица 2 – Конструкторы абстрактного класса State

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | ExpressTrain | Конструктор по умолчанию |  |  |

Таблица 3 – Методы класса

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа | Абстрактный/  неабстрактный | Параметры |
| 1 | OutputInfo() | Метод для вывода информации об обьекте | String | public | неабстрактный |  |

5 Таблица тестов

Таблица 2

|  |  |  |
| --- | --- | --- |
| № | Действие | Результат |
| 1 | Первоначальный  запуск программы | Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход; |
| 2 | Создание нового | Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  2  В какой раздел добавить запись?  1.Автомобиль;  2.Поезд;  3.Экспресс;  0.Выход  1  Модель автомобиля: lexus  Год выпуска:2018  Состояние:В гараже  Средняя скорость:120  Запись добавлена  Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  1  Запись №1:  Автомобиль Модель: Toyota; Год выпуска: 2010; Состояние: В гараже; Средняя скорость: 85;  Запись №2:  Автомобиль Модель: BMW; Год выпуска: 2010; Состояние: В пути; Средняя скорость: 110;  Запись №3:  Поезд Номер поезда: 100; Тип: Грузовой; Количество вагонов: 12; Состояние: В пути; Средняя скорость: 85;  Запись №4:  Поезд Номер поезда: 1; Тип: Почтовый; Количество вагонов: 21; Состояние: Ремонт; Средняя скорость: 90;  Запись №5:  Экспресс-поезд Номер экпресса: 25; Тип: Пассажирский; Количество вагонов: 2; Состояние: В пути; Средняя скорость: 150;  Запись №6:  Экспресс-поезд Номер экпресса: 12; Тип: Грузовой; Количество вагонов: 15; Состояние: В гараже; Средняя скорость: 140;  Запись №7:  Автомобиль Модель: lexus; Год выпуска: 2018; Состояние: В гараже; Средняя скорость: 120; |
| 3 | Поиск заданного | Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  3  1.Поиск по всем данным  2.Поиск по критериям  3.Поиск по номеру записи  0.Выход  1  Что будем искать?  1.Автомобиль;  2.Поезд;  3.Экспресс  1  Модель автомобиля: BMW  Год выпуска:2010  Состояние:В пути  Средняя скорость:110  Запись найдена  Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  3  1.Поиск по всем данным  2.Поиск по критериям  3.Поиск по номеру записи  0.Выход  2  Что будем искать?  1.Автомобиль;  2.Поезд;  3.Экспресс  1  Поиск по  1.Модели автомобиля  2.Году выпуска  3.Состоянию  4.Средней скорости  0.Выход  2  Год выпуска:2010  Запись №1:  Автомобиль Модель: Toyota; Год выпуска: 2010; Состояние: В гараже; Средняя скорость: 85;  Запись №2:  Автомобиль Модель: BMW; Год выпуска: 2010; Состояние: В пути; Средняя скорость: 110;  Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  3  1.Поиск по всем данным  2.Поиск по критериям  3.Поиск по номеру записи  0.Выход  3  Запись №2  Запись найдена  Запись №2:  Автомобиль Модель: BMW; Год выпуска: 2010; Состояние: В пути; Средняя скорость: 110; |
| 4 | Изменение данных | Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  4  Какую запись изменить?  Запись №  2  Изменить состояние на:В пути  Измененный:  Запись №2:  Автомобиль Модель: BMW; Год выпуска: 2010; Состояние: В пути; Средняя скорость: 110; |
| 5 | Вывод списка | Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  1  Запись №1:  Автомобиль Модель: Toyota; Год выпуска: 2010; Состояние: В гараже; Средняя скорость: 85;  Запись №2:  Автомобиль Модель: BMW; Год выпуска: 2010; Состояние: В пути; Средняя скорость: 110;  Запись №3:  Поезд Номер поезда: 100; Тип: Грузовой; Количество вагонов: 12; Состояние: В пути; Средняя скорость: 85;  Запись №4:  Поезд Номер поезда: 1; Тип: Почтовый; Количество вагонов: 21; Состояние: Ремонт; Средняя скорость: 90;  Запись №5:  Экспресс-поезд Номер экпресса: 25; Тип: Пассажирский; Количество вагонов: 2; Состояние: В пути; Средняя скорость: 150;  Запись №6:  Экспресс-поезд Номер экпресса: 12; Тип: Грузовой; Количество вагонов: 15; Состояние: В гараже; Средняя скорость: 140; |
| 6 | Удаление … … | Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  1  Запись №1:  Автомобиль Модель: Toyota; Год выпуска: 2010; Состояние: В гараже; Средняя скорость: 85;  Запись №2:  Автомобиль Модель: BMW; Год выпуска: 2010; Состояние: В пути; Средняя скорость: 110;  Запись №3:  Поезд Номер поезда: 100; Тип: Грузовой; Количество вагонов: 12; Состояние: В пути; Средняя скорость: 85;  Запись №4:  Поезд Номер поезда: 1; Тип: Почтовый; Количество вагонов: 21; Состояние: Ремонт; Средняя скорость: 90;  Запись №5:  Экспресс-поезд Номер экпресса: 12; Тип: Грузовой; Количество вагонов: 15; Состояние: В гараже; Средняя скорость: 140;  Запись №6:  Автомобиль Модель: lexus; Год выпуска: 2018; Состояние: В гараже; Средняя скорость: 120;  Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  5  Какую запись удалить?  Запись №2  Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  1  Запись №1:  Автомобиль Модель: Toyota; Год выпуска: 2010; Состояние: В гараже; Средняя скорость: 85;  Запись №2:  Поезд Номер поезда: 100; Тип: Грузовой; Количество вагонов: 12; Состояние: В пути; Средняя скорость: 85;  Запись №3:  Поезд Номер поезда: 1; Тип: Почтовый; Количество вагонов: 21; Состояние: Ремонт; Средняя скорость: 90;  Запись №4:  Экспресс-поезд Номер экпресса: 12; Тип: Грузовой; Количество вагонов: 15; Состояние: В гараже; Средняя скорость: 140;  Запись №5:  Автомобиль Модель: lexus; Год выпуска: 2018; Состояние: В гараже; Средняя скорость: 120; |
| 7 | Завершение  работы программы | Имеется список транспортных средств компании РЖД,выберите действие:  1.Вывести список;  2.Добавить запись;  3.Найти запись  4.Изменить запись  5.Удалить запись  0.Выход;  0  До скорой встречи! |

6 Результаты тестирования

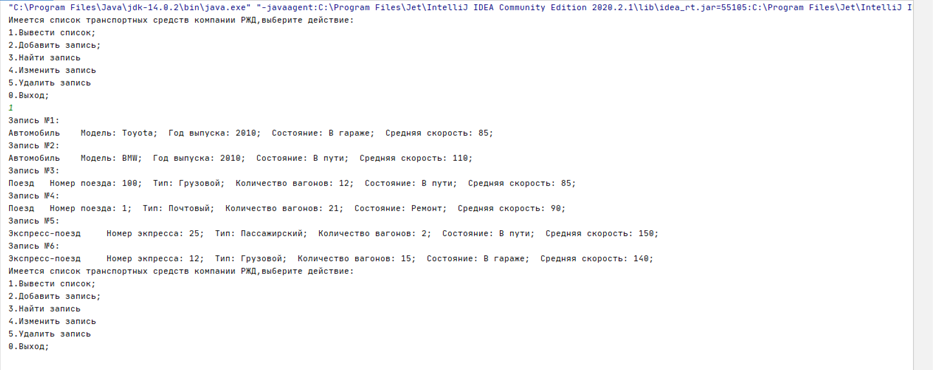


Рисунок 3-результат теста №1 и №5

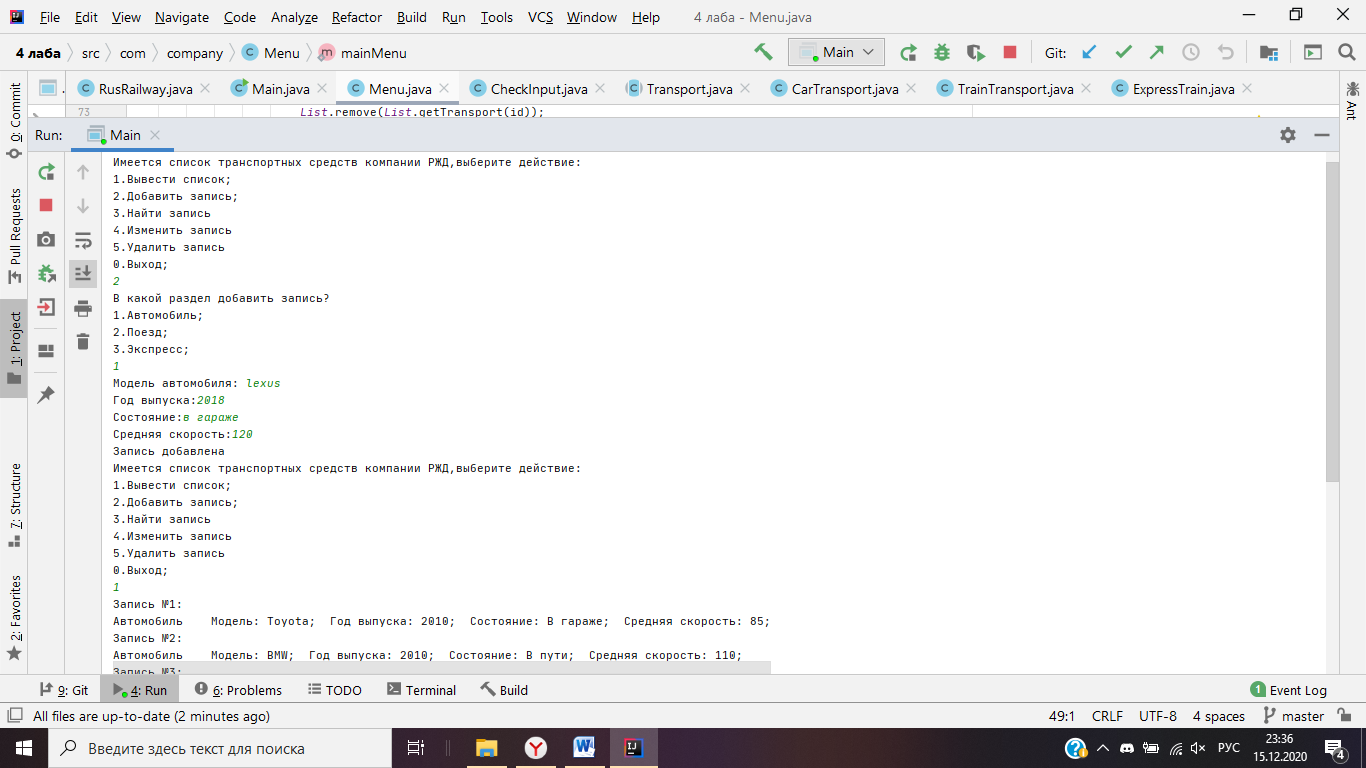


Рисунок 4-результат теста №2

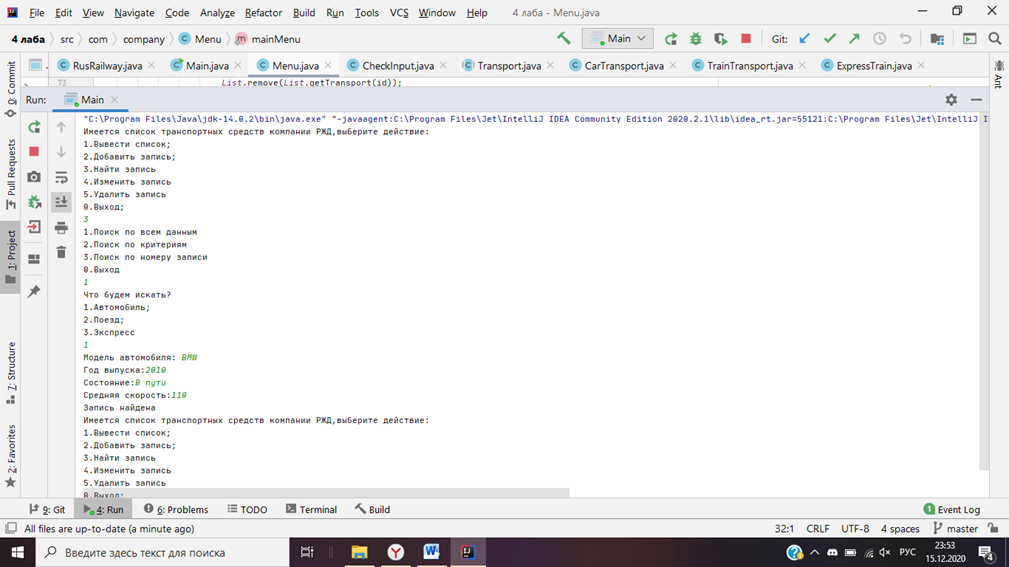


Рисунок 5-результат теста №3

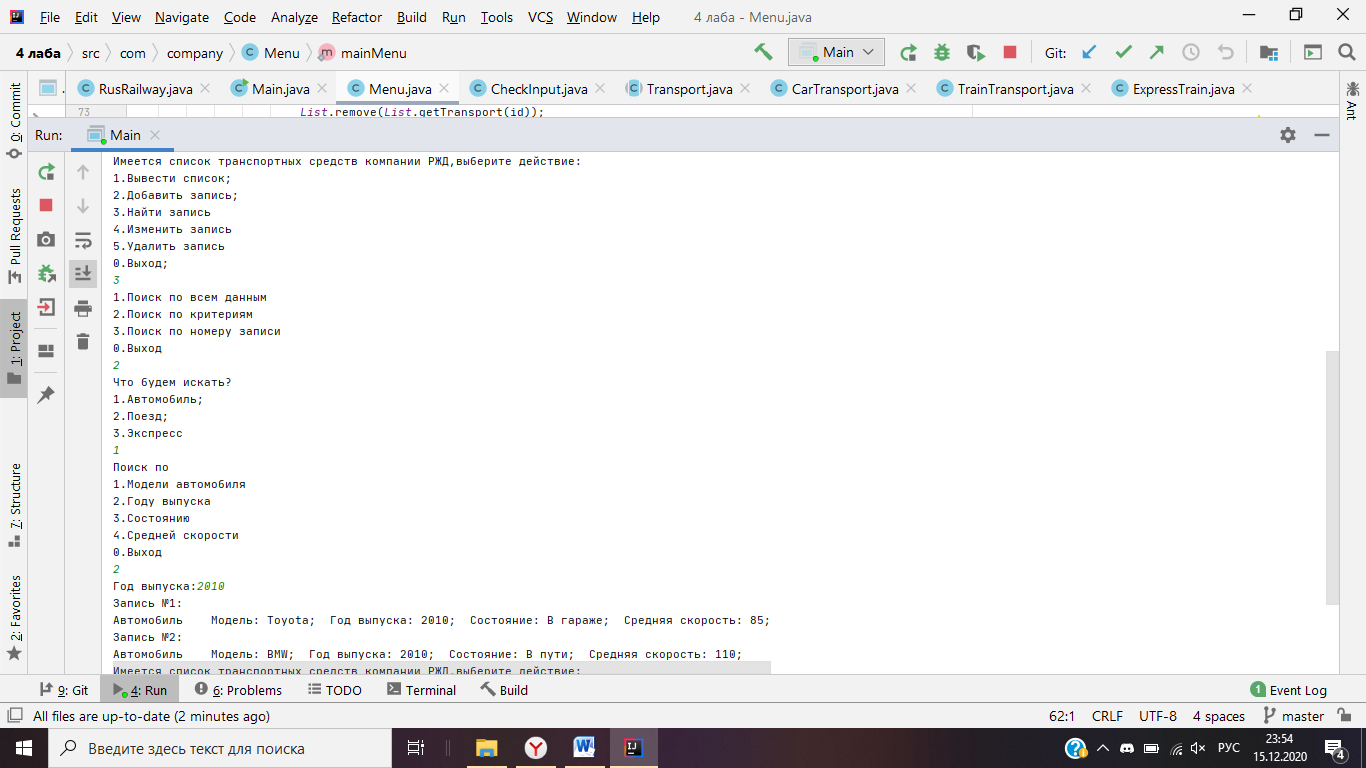


Рисунок 6-результат теста №3

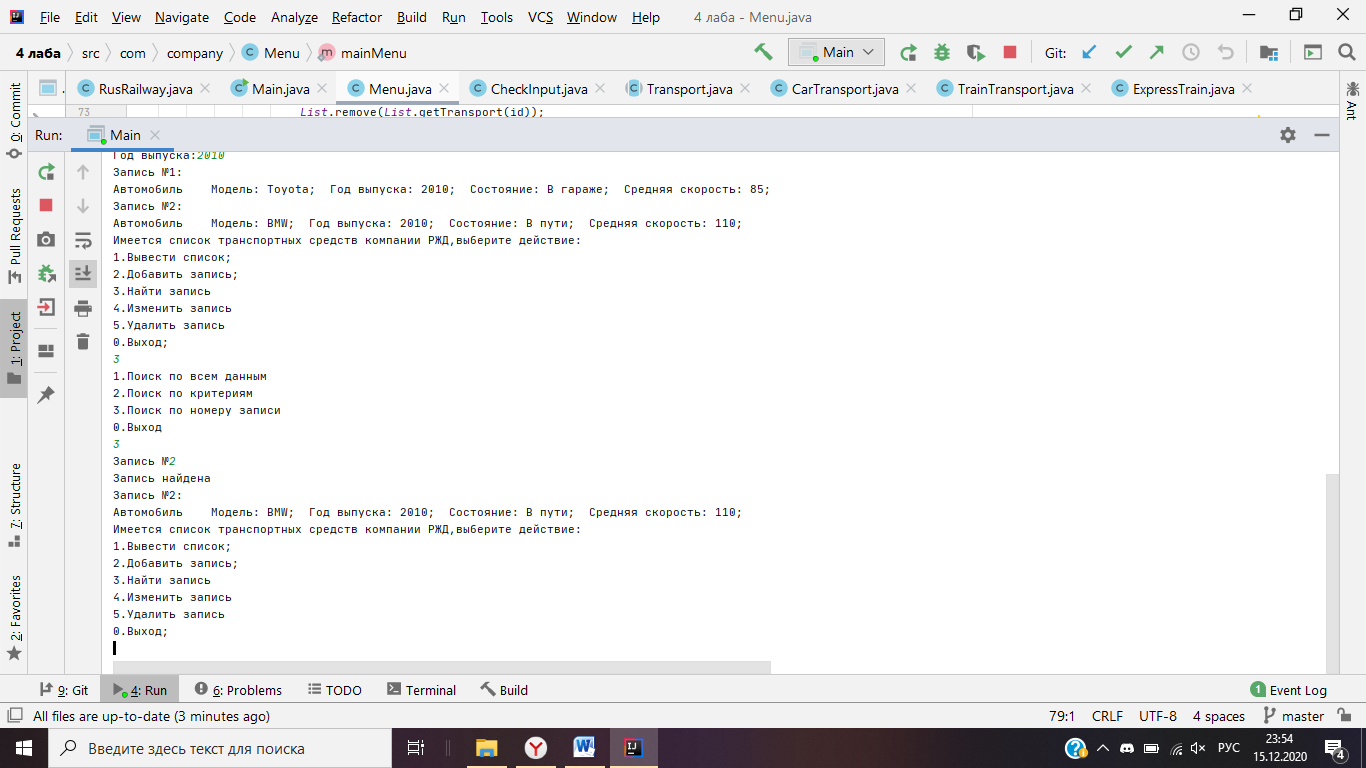


Рисунок 7-результат теста №3

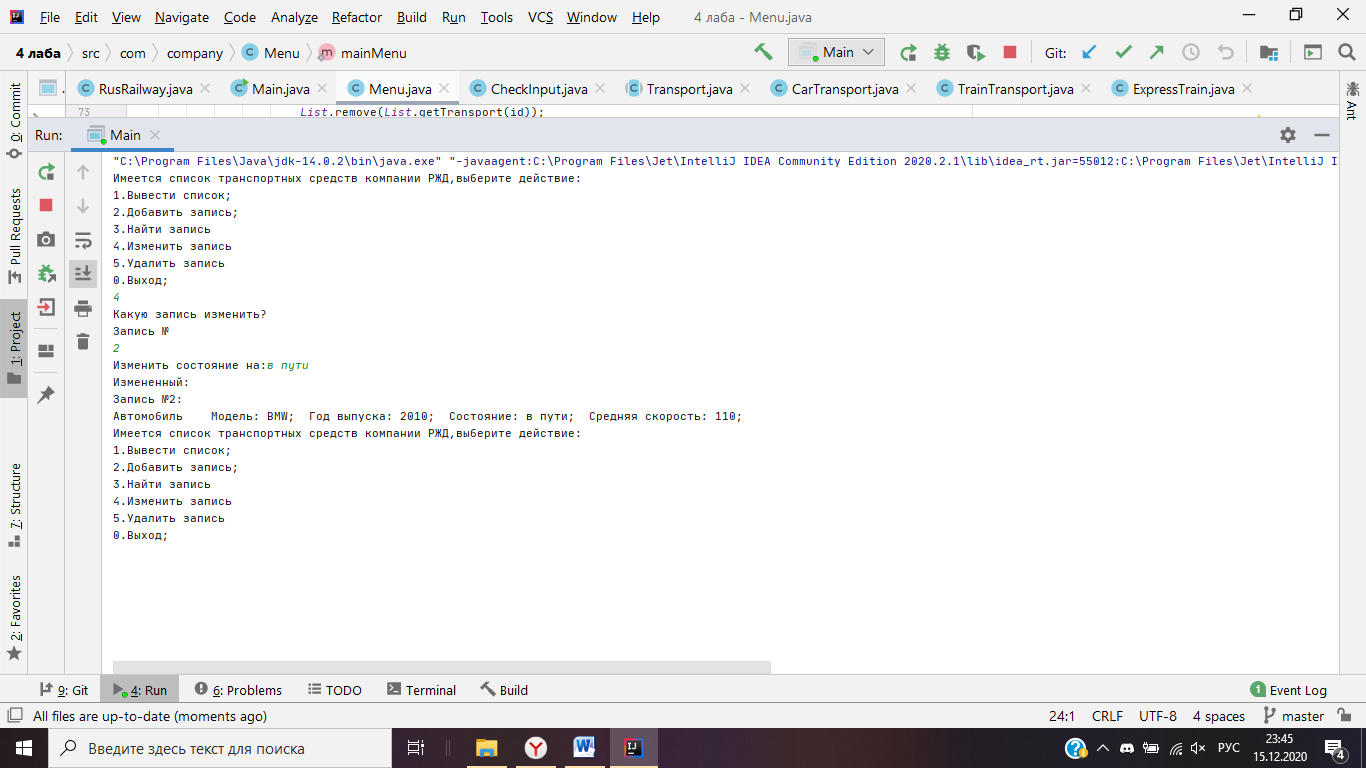


Рисунок 8-результат теста №4

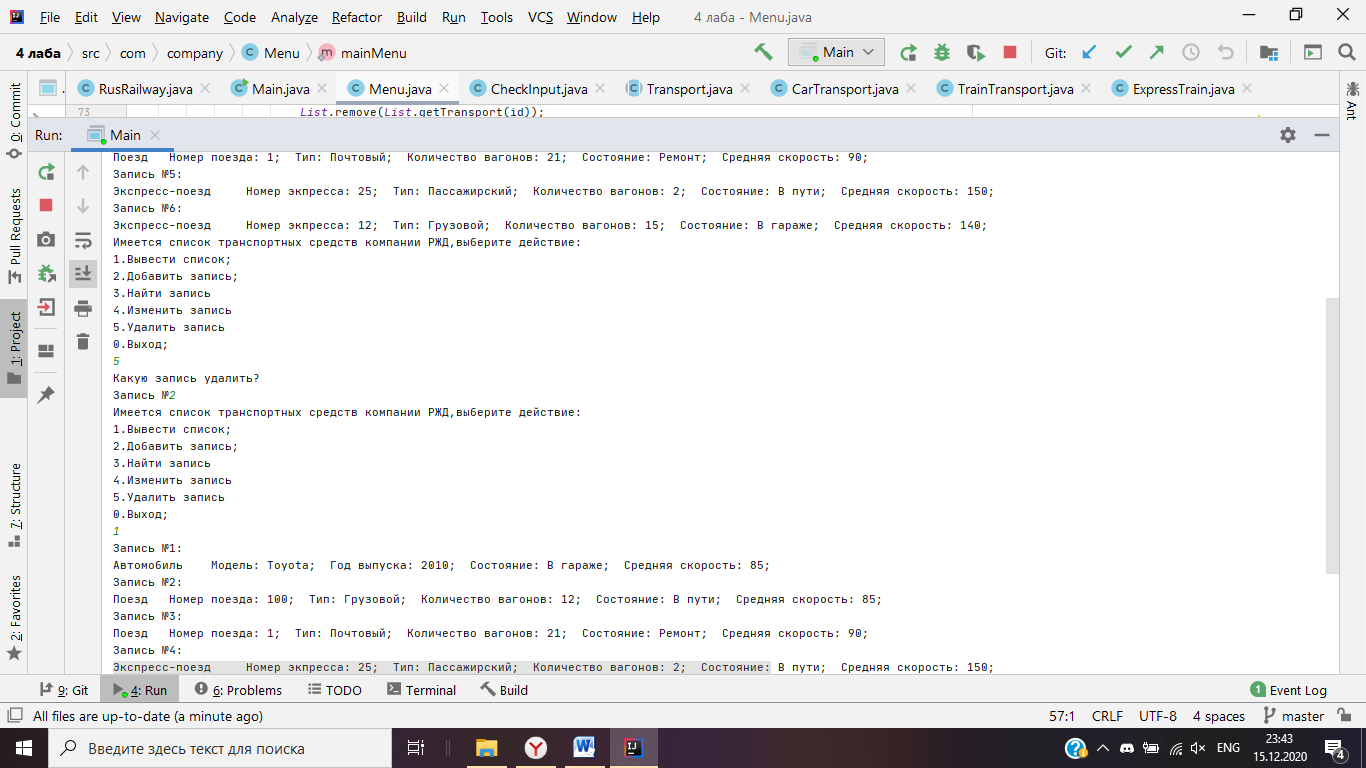


Рисунок 9-результат теста №6

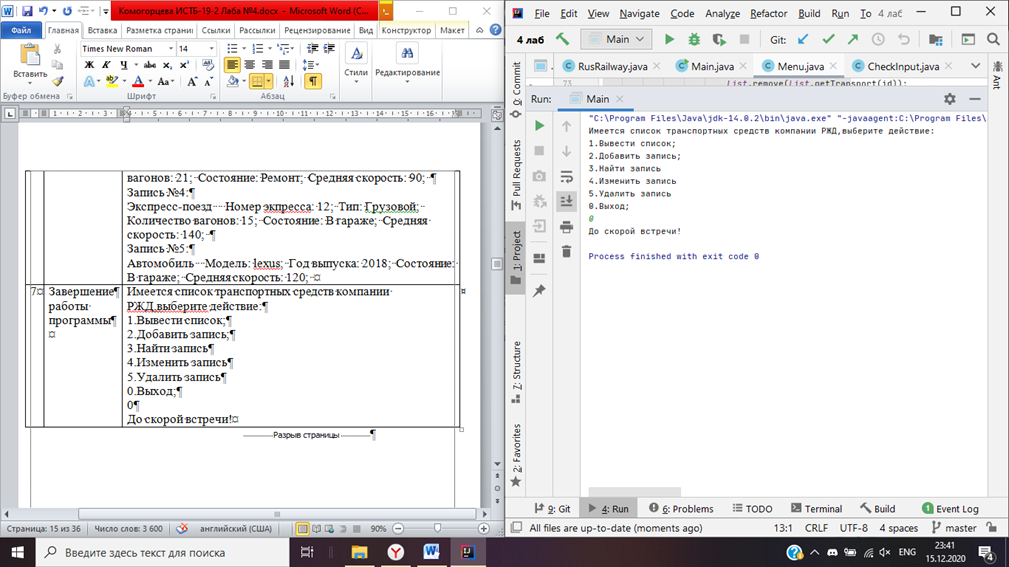


Рисунок 10-результат теста №7

7 Листинг программы

**class Main**

**package com.company;**

**public class Main {**

**public static void main(String[] args) {**

**Menu.mainMenu();**

**}**

**}**

**class Menu**

**package** com.company;  
**public class** Menu {  
 *//наш лист,с которым будем работать* **public static** RusRailway *List* =**new** RusRailway();  
 **public static void** mainMenu() {  
 *//создаем записи  
 List*.add(**new** CarTransport(**"Toyota"**,2010,**"В гараже"**,85));  
 *List*.add(**new** CarTransport(**"BMW"**,2010,**"В пути"**,110));  
 *List*.add(**new** TrainTransport(100 ,**"Грузовой"**,12,**"В пути"**,85));  
 *List*.add(**new** TrainTransport(1 ,**"Почтовый"**,21,**"Ремонт"**,90));  
 *List*.add(**new** ExpressTrain(25 ,**"Пассажирский"**,2,**"В пути"**,150));  
 *List*.add(**new** ExpressTrain(12 ,**"Грузовой"**,15,**"В гараже"**,140));  
 **int** key1;  
 **do** {  
 System.***out***.println(**"Имеется список транспортных средств компании РЖД,выберите действие:"**);  
 System.***out***.println(**"1.Вывести список;"**);  
 System.***out***.println(**"2.Добавить запись;"**);  
 System.***out***.println(**"3.Найти запись"**);  
 System.***out***.println(**"4.Изменить запись"**);  
 System.***out***.println(**"5.Удалить запись"**);  
 System.***out***.println(**"0.Выход;"**);  
 *//проверка что ввели число* key1=CheckInput.*InputInt*();  
 **switch** (key1) {  
 **case** 1:  
 *//вывод листа  
 List*.print();  
 **break**;  
 **case** 2: {  
 *//добавить запись* System.***out***.println(**"В какой раздел добавить запись?"**);  
 System.***out***.println(**"1.Автомобиль;"**);  
 System.***out***.println(**"2.Поезд;"**);  
 System.***out***.println(**"3.Экспресс;"**);  
 **int** key = CheckInput.*InputInt*();  
 **while** (key > 3 || key < 0) {  
 System.***out***.println(**"Вводите только числа 0,1,2,3 .Повторите ввод: "**);  
 key = CheckInput.*InputInt*();  
 }  
 *List*.add(CheckInput.*Input*(key));  
 System.***out***.println(**"Запись добавлена"**);  
 **break**;  
 }  
 **case** 3:  
 *//найти запись  
 search*();  
 **break**;  
 **case** 4: {  
 *//изменить запись* System.***out***.println(**"Какую запись изменить?"**);  
 System.***out***.println(**"Запись №"**);  
 **int** id = CheckInput.*InputInt*();  
 *//если запись ,номер которой ввел пользователь, существует* **if** (id < *List*.size() || (id > 0)) {  
 System.***out***.print(**"Изменить состояние на:"**);  
 *//вызов метода изменения записи(элемент списка ,на что менять)  
 List*.getTransport(id).setCondition(CheckInput.*InputStr*());  
 System.***out***.println(**"Измененный:"** );  
 *List*.print1(id);  
 }  
 **if** (id > *List*.size() || (id < 0)) System.***out***.println(**"Нет такой записи"**);  
 **break**;  
 }  
 **case** 5: {  
 *//удалить запись* System.***out***.println(**"Какую запись удалить?"**);  
 System.***out***.print(**"Запись №"**);  
 *//чтение номера записи* **int** id = CheckInput.*InputInt*();  
 *//если запись,номер которой ввел пользователь, существует* **if** (id <= *List*.size() || (id > 0)) {  
 *//вызов метода удаления записи(элемент списка)  
 List*.remove(*List*.getTransport(id));  
 }  
 **if** (id-1 > *List*.size() || (id <= 0)) System.***out***.println(**"Нет такой записи"**);  
 **break**;  
 }  
 **case** 0:  
 **break**;  
 **default**:  
 System.***out***.println(**"Нет такого пункта"**);  
 }  
 } **while** (key1!=0);  
 System.***out***.println(**"До скорой встречи!"**);  
 }  
 *//метод для поиска записи* **private static void** search(){  
 *//какой критерий,выражается цифрой* **int** whatCriterion = 0;  
 *//выражение по которому искать* String criterion = **null**;  
 System.***out***.println(**"1.Поиск по всем данным"**);  
 System.***out***.println(**"2.Поиск по критериям"**);  
 System.***out***.println(**"3.Поиск по номеру записи"**);  
 System.***out***.println(**"0.Выход"**);  
 **switch** (CheckInput.*InputInt*()) {  
 **case** 1: {  
 System.***out***.println(**"Что будем искать?"**);  
 System.***out***.println(**"1.Автомобиль;"**);  
 System.***out***.println(**"2.Поезд;"**);  
 System.***out***.println(**"3.Экспресс"**);  
 *//проверка на ввод* **int** key=CheckInput.*InputInt*();  
 **while** (key>3||key<1){  
 System.***out***.println(**"Вводите только числа 1,2,3 .Повторите ввод: "**);  
 key=CheckInput.*InputInt*();  
 }  
 *//вызывается метод проверки есть ли такая запись в листе* **if** (*List*.search(key,CheckInput.*Input*(key))) {  
 System.***out***.println(**"Запись найдена"**);  
 }  
 **else** System.***out***.println(**"Запись не найдена"**);  
 **break**;  
 }  
 **case** 2: {  
 System.***out***.println(**"Что будем искать?"**);  
 System.***out***.println(**"1.Автомобиль;"**);  
 System.***out***.println(**"2.Поезд;"**);  
 System.***out***.println(**"3.Экспресс"**);  
 **int** key=CheckInput.*InputInt*();  
 **while** (key>3||key<1){  
 System.***out***.println(**"Вводите только числа 1,2,3 .Повторите ввод: "**);  
 key=CheckInput.*InputInt*();  
 }  
 *//выбор по критериям* **switch** (key) {  
 **case** 1: {  
 System.***out***.println(**"Поиск по"**);  
 System.***out***.println(**"1.Модели автомобиля"**);  
 System.***out***.println(**"2.Году выпуска"**);  
 System.***out***.println(**"3.Состоянию"**);  
 System.***out***.println(**"4.Средней скорости"**);  
 System.***out***.println(**"0.Выход"**);  
 **int** key2 = CheckInput.*InputInt*();  
 *//выбор поиска по критериям для автомобиля  
 //переменная whatCriterion от 1 до 4 ,каждая цифра для своего критерия* **switch** (key2) {  
 **case** 1: {  
 System.***out***.print(**"Модель автомобиля:"**);  
 whatCriterion = 1;  
 **break**;  
 }  
 **case** 2: {  
 System.***out***.print(**"Год выпуска:"**);  
 whatCriterion = 2;  
 **break**;  
 }  
 **case** 3: {  
 System.***out***.print(**"Состояние:"**);  
 whatCriterion = 3;  
 **break**;  
 }  
 **case** 4: {  
 System.***out***.print(**"Средняя скорость:"**);  
 whatCriterion = 4;  
 **break**;  
 }  
 **case** 0:  
 **break**;  
 **default**:  
 System.***out***.println(**"Нет такого пункта"**);  
 }  
 **break**;  
 }  
 **case** 2: {  
 System.***out***.println(**"Поиск по"**);  
 System.***out***.println(**"1.Номеру поезда"**);  
 System.***out***.println(**"2.Типу"**);  
 System.***out***.println(**"3.Количеству вагонов"**);  
 System.***out***.println(**"4.Состоянию"**);  
 System.***out***.println(**"5.Средней скорости"**);  
 System.***out***.println(**"0.Выход"**);  
 **int** key2 = CheckInput.*InputInt*();  
 *//выбор поиска по критериям для поезда  
 //переменная whatCriterion от 1 до 5 ,каждая цифра для своего критерия* **switch** (key2) {  
 **case** 1: {  
 System.***out***.print(**"Номер поезда:"**);  
 whatCriterion = 1;  
 **break**;  
 }  
 **case** 2: {  
 System.***out***.print(**"Тип:"**);  
 whatCriterion = 2;  
 **break**;  
 }  
 **case** 3: {  
 System.***out***.print(**"Количество вагонов:"**);  
 whatCriterion = 3;  
 **break**;  
 }  
 **case** 4: {  
 System.***out***.print(**"Состояние:"**);  
 whatCriterion = 4;  
 **break**;  
 }  
 **case** 5: {  
 System.***out***.print(**"Средняя скорость:"**);  
 whatCriterion = 5;  
 **break**;  
 }  
 **case** 0:  
 **break**;  
 **default**:  
 System.***out***.println(**"Нет такого пункта"**);  
 }  
 **break**;  
 }  
 **case** 3: {  
 System.***out***.println(**"Поиск по"**);  
 System.***out***.println(**"1.Номеру экспресса"**);  
 System.***out***.println(**"2.Типу"**);  
 System.***out***.println(**"3.Количеству вагонов"**);  
 System.***out***.println(**"4.Состоянию"**);  
 System.***out***.println(**"5.Средней скорости"**);  
 System.***out***.println(**"0.Выход"**);  
 **int** key2 = CheckInput.*InputInt*();  
 *//выбор поиска по критериям для экспресса  
 //переменная whatCriterion от 1 до 5 ,каждая цифра для своего критерия* **switch** (key2) {  
 **case** 1: {  
 System.***out***.print(**"Номер экспресса:"**);  
 whatCriterion = 1;  
 **break**;  
 }  
 **case** 2: {  
 System.***out***.print(**"Тип:"**);  
 whatCriterion = 2;  
 **break**;  
 }  
 **case** 3: {  
 System.***out***.print(**"Количество вагонов:"**);  
 whatCriterion = 3;  
 **break**;  
 }  
 **case** 4: {  
 System.***out***.print(**"Состояние:"**);  
 whatCriterion = 4;  
 **break**;  
 }  
 **case** 5: {  
 System.***out***.print(**"Средняя скорость:"**);  
 whatCriterion = 5;  
 **break**;  
 }  
 **case** 0:  
 **break**;  
 **default**:  
 System.***out***.println(**"Нет такого пункта"**);  
 }  
 **break**;  
 }  
 **default**:  
 System.***out***.println(**"Нет такого пункта"**);  
 }  
 *//в свитчах нашли по какому критерию будем искать  
 //далее считываем что будем искать в найденном критерии* criterion = CheckInput.*InputStr*();  
 *//вызывается метод поиска по критерию (ключ-что ищем машина,поезд,экпресс),какой критерий,что быдем искать в данном критерии)* **if** (*List*.searchCriterion(key, whatCriterion, criterion) == **false**) {  
 System.***out***.println(**"Записи не найдены"**);  
 }  
 **break**;  
 }  
 **case** 3: {  
 *//поиск по номеру записи* System.***out***.print(**"Запись №"**);  
 **int** id = CheckInput.*InputInt*();  
 *//если такая запись существует,то она найдена,и выводим ее* **if**(id<= *List*.size()&(id>0)) {  
 System.***out***.println(**"Запись найдена"**);  
 *List*.print1(id);  
 }  
 **else** System.***out***.println(**"Нет такой записи"**);  
 **break**;  
 }  
 **case** 0:  
 **break**;  
 **default**:  
 System.***out***.println(**"Нет такого пункта"**);  
 }  
 }  
  
}

**class CheckInput**

**package** com.company;  
**import** java.util.Scanner;  
*//класс для проверки ввода***public class** CheckInput {  
 **private static** Scanner *in* = **new** Scanner(System.***in***);  
 *//метод для ввода "обьекта абстрактного класса Transport"  
 //но это невозможно,а имеется в виду для ввода детей и внуков класса Transport* **public static** Transport Input(**int** key) {  
 **switch** (key) {  
 **case** 1: {  
 *//ввод обьекта класса CarTransport* System.***out***.print(**"Модель автомобиля: "**);  
 String model = CheckInput.*InputStr*();  
 System.***out***.print(**"Год выпуска:"**);  
 **int** years = CheckInput.*InputInt*();  
 System.***out***.print(**"Состояние:"**);  
 String condition = CheckInput.*InputStr*();  
 System.***out***.print(**"Средняя скорость:"**);  
 **int** speed = CheckInput.*InputInt*();  
 **return** (**new** CarTransport(model, years, condition, speed));  
 }  
 **case** 2: {  
 *//ввод обьекта класса TrainTransport* System.***out***.print(**"Номер поезда:"**);  
 **int** number = CheckInput.*InputInt*();  
 System.***out***.print(**"Тип: "**);  
 String type = CheckInput.*InputStr*();  
 System.***out***.print(**"Количество вагонов:"**);  
 **int** carriage = CheckInput.*InputInt*();  
 System.***out***.print(**"Состояние:"**);  
 String condition = CheckInput.*InputStr*();  
 System.***out***.print(**"Средняя скорость:"**);  
 **int** speed = CheckInput.*InputInt*();  
 **return** (**new** TrainTransport(number, type, carriage, condition, speed));  
 }  
 **case** 3:  
 {  
 *//ввод обьекта класса ExpressTrain* System.***out***.print(**"Номер экпресса:"**);  
 **int** number = *InputInt*();  
 System.***out***.print(**"Тип: "**);  
 String type = *InputStr*();  
 System.***out***.print(**"Количество вагонов:"**);  
 **int** carriage = *InputInt*();  
 System.***out***.print(**"Состояние:"**);  
 String condition = *InputStr*();  
 System.***out***.print(**"Средняя скорость:"**);  
 **int** speed = *InputInt*();  
 **return** (**new** ExpressTrain(number, type, carriage, condition, speed));  
 }  
 }  
 **return null**;  
 }  
 *//метод для проверки что в консоли ввели число* **public static int** InputInt(){  
 **int** read ;  
 **try**{  
 read = Integer.*parseInt*(*in*.nextLine());  
 }**catch** (NumberFormatException ex){  
 System.***out***.print(**"Вводите только числа,пожалуйста.Повторите ввод: "**);  
 read=*InputInt*();  
 }  
 **return** read;  
 }  
 *//метод для проверки что в консоли ввели строку* **public static** String InputStr(){  
 String rez=**null**;  
 **try**{  
 rez=*in*.nextLine();  
 **int** i=1/rez.length();  
 }  
 **catch** (ArithmeticException e) {  
 System.***out***.println(**"Вы ничего не ввели.Повторите ввод:"**);  
 *InputStr*();  
 }  
 **return** rez;  
 }  
  
}

**class RusRailway**

**package** com.company;  
**import** java.util.ArrayList;  
*//класс для работы с динамическим массивом "Транспорт компании РЖД"***public class** RusRailway {  
 **private final static** ArrayList<Transport> ***RusRailwayTransports***=**new** ArrayList();  
 *//добавление записи в лист/массив* **public void** add(Transport transport) {  
 **this**.***RusRailwayTransports***.add(transport);  
 }  
 *//вывод списка/массива/листа* **public void** print() {  
 **int** i=1;  
 **for** (**var** transport : ***RusRailwayTransports***)  
 {  
 System.***out***.println(**"Запись №"**+i+**":\n"**+transport.OutputInfo());  
 i++;  
 }  
 }  
 *//вывод одного элемента списка/массива листа* **public void** print1(**int** id) {  
 **for** (**var** transport : ***RusRailwayTransports***) {  
 **if** (transport.getId()==id){  
 System.***out***.println(**"Запись №"**+id+**":\n"**+***RusRailwayTransports***.get(id-1).OutputInfo());  
 }  
 }  
 }  
 *//геттер* **public** Transport getTransport(**int** id) {  
 **for** (Transport tr:***RusRailwayTransports***) {  
 **if** (tr.getId()==id){  
 **return** tr;  
 }  
 }  
 **return null**;  
 }  
 *//метод для удаления записи* **public void** remove(Transport transport){  
 **this**.***RusRailwayTransports***.remove(transport);  
 }  
 *//метод для поиска по целому обьекту* **public boolean** search(**int** key,Transport transp) {  
 **boolean** rez = **false**;  
 *//перебор записей листа* **for** (**var** transport : ***RusRailwayTransports***) {  
 **switch** (key){  
 **case** 1:  
 *//если элемент списка принадлежит классу CarTransport* **if** (transport **instanceof** CarTransport) {  
 *//если элемент списка совпадает с элементом которых запрашивали* **if** (transport.getCondition().equals(transp.getCondition()) &&  
 transport.getSpeed() == transp.getSpeed() &&  
 ((CarTransport) transport).getYears() == ((CarTransport) transp).getYears() &&  
 ((CarTransport) transport).getModel().equals(((CarTransport) transp).getModel())) {  
 *//возвращаем правду* rez = **true**;  
 }  
 }  
 **break**;  
 **case** 2:  
 *//если элемент списка принадлежит классу TrainTransport* **if** (transport **instanceof** TrainTransport) {  
 *//если элемент списка совпадает с элементом которых запрашивали* **if** (((TrainTransport) transport).getNumber() == ((TrainTransport) transp).getNumber() &&  
 ((TrainTransport) transport).getType().equals(((TrainTransport) transp).getType()) &&  
 ((TrainTransport) transport).getCarriage() == ((TrainTransport) transp).getCarriage() &&  
 transport.getCondition().equals(transp.**condition**) && transport.getSpeed() == transp.getSpeed()) {  
 *//возвращаем правду* rez = **true**;  
 }  
 }  
 **break**;  
 **case** 3:  
 *//если элемент списка принадлежит классу ExpressTrain* **if** (transport **instanceof** ExpressTrain) {  
 *//если элемент списка совпадает с элементом которых запрашивали* **if** (((ExpressTrain) transport).getNumber() == ((ExpressTrain) transp).getNumber() &&  
 ((ExpressTrain) transport).getType().equals(((ExpressTrain) transp).getType()) &&  
 ((ExpressTrain) transport).getCarriage() == ((ExpressTrain) transp).getCarriage() &&  
 transport.getCondition().equals(transp.getCondition()) &&  
 transport.getSpeed() == transp.getSpeed()) {  
 *//возвращаем правду* rez = **true**;  
 }  
 }  
 **break**;  
 }  
 }  
 *//возвращаем "правду" если нашли, и "ложь" если не нашли* **return** rez;  
 }  
 *//метод для поиска по отдельным критериям* **public boolean** searchCriterion(**int** key,**int** whatCriterion,String criterion){  
 **boolean** result=**false**;  
 *//перебор записей листа* **for** (**var** transport : ***RusRailwayTransports***) {  
 **switch** (key){  
 **case** 1:  
 {  
 *//если элемент списка принадлежит классу CarTransport  
 //то находим нужный нам критерий для поиска  
 // сравниваем есть ли по этому критерию элемент с нужным значением критерия  
 //и выводим его* **if** (transport **instanceof** CarTransport) {  
 **if** (whatCriterion == 1) {  
 **if** (((CarTransport) transport).getModel().equals(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 2) {  
 **if** (((CarTransport) transport).getYears() ==Integer.*parseInt*(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 3) {  
 **if** (transport.getCondition().equals(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 4) {  
 **if** (transport.getSpeed() == Integer.*parseInt*(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 }  
 }  
 **break**;  
 **case** 2:  
 {  
 *//если элемент списка принадлежит классу TrainTransport  
 //то находим нужный нам критерий для поиска  
 // сравниваем есть ли по этому критерию элемент с нужным значением критерия  
 //и выводим его* **if** (transport **instanceof** TrainTransport) {  
 **if** (whatCriterion == 1) {  
 **if** (((TrainTransport) transport).getNumber()==Integer.*parseInt*(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 2) {  
 **if** (((TrainTransport) transport).getType().equals(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 3) {  
 **if** (((TrainTransport) transport).getCarriage()==Integer.*parseInt*(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 4) {  
 **if** (transport.getCondition().equals(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 5) {  
 **if** (transport.getSpeed() == Integer.*parseInt*(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 }  
 }  
 **break**;  
 **case** 3:  
 {  
 *//если элемент списка принадлежит классу ExpressTrain  
 //то находим нужный нам критерий для поиска  
 // сравниваем есть ли по этому критерию элемент с нужным значением критерия  
 //и выводим его* **if** (transport **instanceof** ExpressTrain) {  
 **if** (whatCriterion == 1) {  
 **if** (((ExpressTrain) transport).getNumber()==Integer.*parseInt*(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 2) {  
 **if** (((ExpressTrain) transport).getType().equals(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 3) {  
 **if** (((ExpressTrain) transport).getCarriage()==Integer.*parseInt*(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 4) {  
 **if** (transport.getCondition().equals(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 **else  
 if** (whatCriterion == 5) {  
 **if** (transport.getSpeed() == Integer.*parseInt*(criterion)) {  
 result = **true**;  
 print1(transport.getId());  
 }  
 }  
 }  
 }  
 **break**;  
 }  
 }  
 *//возвращаем "правду" если нашли, и "ложь" если не нашли* **return** result;  
 }  
 *//метод возвращающий размер листа* **public int** size() {  
 **return this**.***RusRailwayTransports***.size();  
 }  
}

class Transport

**package** com.company;  
*//абстрактный класс(родитель)***public abstract class** Transport {  
 *//поля и метод абстрактного класса,их наследуют дети и внуки* **protected** String **condition**;  
 **protected int speed**;  
 **public abstract** String OutputInfo();  
 **protected int id**;  
 *//конструктор по умолчанию* **public** Transport(){  
 **this**.**id**=*count*;  
 *count*++;  
 }  
 **private static int** *count*=1;  
 *//геттеры и сеттеры* **public int** getId() {  
 **return id**;  
 }  
 **public int** getSpeed() {  
 **return speed**;  
 }  
 **public** String getCondition() {  
 **return condition**;  
 }  
 **public void** setCondition(String condition) {  
 **this**.**condition** = condition;  
 }  
  
}

class CarTransport

**package** com.company;  
*//класс -ребенок по отношению к классу Transport***public class** CarTransport **extends** Transport {  
 *//поля класса CarTransport* **protected int years**;  
 **protected** String **model**;  
 *//геттеры* **public** String getModel() {  
 **return model**;  
 }  
 **public int** getYears() {  
 **return years**;  
 }  
 *//конструктор с пармаметром* **public** CarTransport(String model, **int** years, String condition, **int** speed) {  
 **this**.**model**=model;  
 **this**.**years**=years;  
 **this**.**condition**=condition;  
 **this**.**speed**=speed;  
 }  
 *//переопределенный метод для вывода информации* @Override  
 **public** String OutputInfo() {  
 StringBuilder str=**new** StringBuilder(**"Автомобиль "**);  
 str.append(**"Модель: "**+**this**.getModel()+**"; "**);  
 str.append(**"Год выпуска: "**+**this**.getYears()+**"; "**);  
 str.append(**"Состояние: "**+**this**.getCondition()+**"; "**);  
 str.append(**"Средняя скорость: "**+**this**.getSpeed()+**"; "**);  
 **return** str.toString();  
 }  
}

class TrainTransport

**package** com.company;  
*//класс -ребенок по отношению к классу Transport***public class** TrainTransport **extends** Transport {  
 *//поля класса TrainTransport* **protected** String **type**;  
 **protected int carriage**;  
 **protected int number**;  
 *//конструктор с параметром* **public** TrainTransport(**int** number,String type,**int** carriage,String condition,**int** speed){  
 **this**.**number**=number;  
 **this**.**type**=type;  
 **this**.**carriage**=carriage;  
 **this**.**condition**=condition;  
 **this**.**speed**=speed;  
 }  
 *//геттеры* **public** String getType() {  
 **return type**;  
 }  
 **public int** getNumber() {  
 **return number**;  
 }  
 **public int** getCarriage() {  
 **return carriage**;  
 }  
  
 *//переопределенный метод для вывода информации* @Override  
 **public** String OutputInfo() {  
 StringBuilder str=**new** StringBuilder(**"Поезд "**);  
 str.append(**"Номер поезда: "**+**this**.getNumber()+**"; "**);  
 str.append(**"Тип: "**+**this**.getType()+**"; "**);  
 str.append(**"Количество вагонов: "**+**this**.getCarriage()+**"; "**);  
 str.append(**"Состояние: "**+**this**.getCondition()+**"; "**);  
 str.append(**"Средняя скорость: "**+**this**.getSpeed()+**"; "**);  
 **return** str.toString();  
 }  
}

class ExpressTrain

**package** com.company;  
*//класс внук по отношению к классу Transport  
//класс-ребенок по отношению к классу TrainTransport***public class** ExpressTrain **extends** TrainTransport {  
 *//все поля наслудуются от TrainTransport  
 //конструктор с параметром* **public** ExpressTrain(**int** number, String type, **int** carriage, String condition, **int** speed) {  
 **super**(number, type, carriage, condition, speed);  
 }  
 *//переопределенный метод для вывода информации* @Override  
 **public** String OutputInfo() {  
 StringBuilder str=**new** StringBuilder(**"Экспресс-поезд "**);  
 str.append(**"Номер экпресса: "**+**this**.getNumber()+**"; "**);  
 str.append(**"Тип: "**+**this**.getType()+**"; "**);  
 str.append(**"Количество вагонов: "**+**this**.getCarriage()+**"; "**);  
 str.append(**"Состояние: "**+**this**.getCondition()+**"; "**);  
 str.append(**"Средняя скорость: "**+**this**.getSpeed()+**"; "**);  
 **return** str.toString();  
 }  
  
}

Список использованных источников

1. Объектно-ориентированное программирование. Метод. указания по

выполнению лабораторных работ / сост.: В.Л. Аршинский. – Иркутск : Изд-во

ИРНИТУ, 2017. – 24 c.