Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

|  |
| --- |
| Институт информационных технологий и анализа данных |

наименование института

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

по дисциплине:

|  |
| --- |
| **ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ Программирование** |
| **«Разработка класса с использованием механизмов наследования,** |
| **полиморфизма и инкапсуляции»** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил | АСУб-20-2 |  |  |  | Арбакова А.В. |
|  | шифр группы |  | подпись |  | Фамилия И.О. |
| Проверил |  |  |  |  | Бакшеева Е.Н. |
|  | должность |  | подпись |  | Фамилия И.О. |

Иркутск 2021 г.

**Содержание**

[1. Постановка задачи 3](#_Toc134745637)

[2. Проектирование классов 3](#_Toc134745638)

[3. Описание структуры пользовательского меню 7](#_Toc134745639)

[4. Таблица тестов 8](#_Toc134745640)

[5. Результаты тестирования 11](#_Toc134745641)

[6. Исходный код 14](#_Toc134745642)

[7. Список использованных источников 19](#_Toc134745643)

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 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.

Цель работы:

1. Изучение механизмов наследования, полиморфизма и инкапсуляции на практике.
2. Получение практических навыков описания иерархии классов и организации работы с объектами-членами классов-«групп».
3. Получение практических навыков использования абстрактных классов и переопределения унаследованных методов.
4. **Проектирование классов**

Таблица 1 – Классы.

|  |  |
| --- | --- |
| lab4 | Главный класс, в котором выполняется задание |
| Car | Абстрактный класс-родитель |
| Menu | Класс-меню |
| BMV | Класс наследник |
| AUDI | Класс наследник |
| Toyota | Класс наследник |

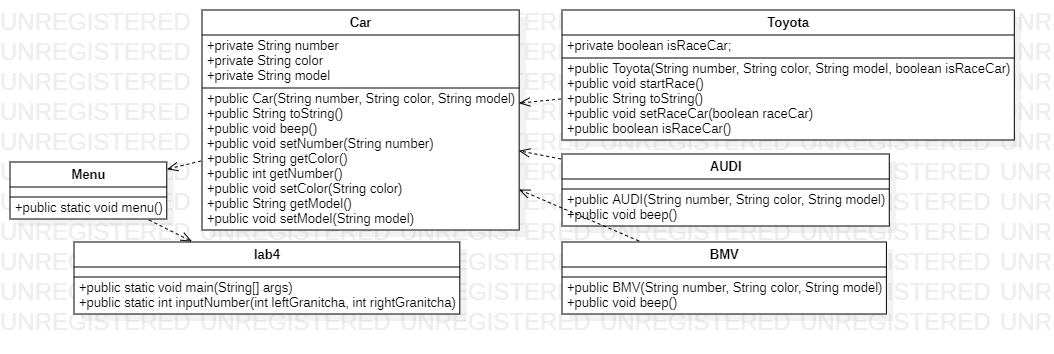
****

Таблица 2 – Таблицы классов, полей и методов.

|  |  |  |  |
| --- | --- | --- | --- |
| **Класс** | **Поля** | **Методы** | **Параметры** |
| lab4 | - | main(String[] args) | String[] args – аргументы переданные через консоль |
| inputNumber(int leftGranitcha, int rightGranitcha) | int leftGranitcha – левая границ диапазона  int rightGranitcha – правая граница диапазона |
| Menu | - | menu () | - |
| Car | String number  String color  String model | Car(String number, String color, String model) | String number – номер машины  String color – цвет машины  String model – модель машины |
| String toString() | - |
| void beep() | - |
| int getNumber() | - |
| void setNumber(String number) | String number – номер машины |
| String getColor() | - |
| void setColor(String color) | String color – цвет машины |
| String getModel() | - |
| void setModel(String model) | String model – модель машины |
| BMV | - | BMV(String number, String color, String model) | String number – номер машины  String color – цвет машины  String model – модель машины |
| void beep() | - |
| AUDI | - | AUDI(String number, String color, String model) | String number – номер машины  String color – цвет машины  String model – модель машины |
| void beep() | - |
| Toyota | boolean isRaceCar | Toyota(String number, String color, String model, boolean isRaceCar) | String number – номер машины  String color – цвет машины  String model – модель машины  boolean isRaceCar – обозначение гоночной машины |
| void startRace() | - |
| String toString() | - |
| void setRaceCar(boolean raceCar) | boolean raceCar – гоночная машина |
| boolean isRaceCar() | - |

Таблица 3 – Спецификации переменных.

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Назначение** |
| 1 | car1 | Car | Объект класса |
| 2 | car2 | Car | Объект класса |
| 3 | car3 | Car | Объект класса |
| 4 | car4 | Car | Объект класса |
| 5 | car5 | Car | Объект класса |
| 6 | car6 | Car | Объект класса |
| 7 | car7 | Car | Объект класса |
| 8 | carList | List <Car> | Список машин |
| 9 | number | String | Номер машины |
| 10 | color | String | Цвет машины |
| 11 | model | String | Модель машины |
| 12 | isRaceCar | boolean | Является гоночной машиной |
| 13 | raceCar | boolean | Является гоночной машиной |
| 14 | leftGranitcha | int | Левая граница диапазона |
| 15 | rightGranitcha | int | Правая граница диапазона |
| 16 | input | String | Переменная для ввода значений |
| 17 | x | int | Дополнительная переменная |
| 18 | isCorrect | boolean | Состояние проверки корректности ввода |
| 19 | console | Scanner | Поток ввода |
| 20 | builder | StringBuilder | Выводит на экран последовательность символов |
| 21 | isWork | boolean | Состояние программы |
| 22 | scanner | Scanner | Поток ввода |
| 23 | i | String | Ввод пункта меню |
| 24 | j | int | Переменная цикла |
| 25 | marka | String | Марка машины |
| 26 | car | Car | Объект класса |
| 27 | colorSearch | String | Переменная для поиска в списке |
| 28 | carInList | List <Car> | Машины в списке |
| 29 | classInfo | String[] | Информация о марке машины |

Таблица 4 – Спецификации функций.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Назначение** | **Параметры** |
| 1 | main (String[] args) | void | Главная функция приложения | String[] args – аргументы переданные через консоль |
| 2 | inputNumber (int leftGranitcha, int rightGranitcha) | int | Проверка вводимого числа | int leftGranitcha – левая границ диапазона  int rightGranitcha – правая граница диапазона |
| 3 | menu () | void | Меню | - |
| 4 | Car(String number, String color, String model) | - | Конструктор Car | String number – номер машины  String color – цвет машины  String model – модель машины |
| 5 | BMV(String number, String color, String model) | - | Конструктор BMV | String number – номер машины  String color – цвет машины  String model – модель машины |
| 6 | AUDI(String number, String color, String model) | - | Конструктор AUDI | String number – номер машины  String color – цвет машины  String model – модель машины |
| 7 | Toyota(String number, String color, String model, boolean isRaceCar) | - | Конструктор Toyota | String number – номер машины  String color – цвет машины  String model – модель машины  isRaceCar – является гоночной машиной |
| 8 | toString() | String | Вывод параметров машины | - |
| 9 | beep() | void | Функция звука машины | - |
| 10 | getNumber() | int | Getter для получения номера машины | - |
| 11 | setNumber(int number) | void | Setter для записи номера машины | String number – номер машины |
| 12 | getColor() | String | Getter для получения цвета машины | - |
| 13 | setColor(String color) | void | Setter для записи цвета машины | String color – цвет машины |
| 14 | getModel() | String | Getter для получения модели машины | - |
| 15 | setModel(String model) | void | Setter для записи модели машины | String model – модель машины |
| 16 | startRace() | void | Функция «начала гонки» | - |
| 17 | setRaceCar(boolean raceCar) | void | Setter для записи параметра гоночной машины | boolean raceCar – гоночная машина |
| 18 | isRaceCar() | boolean | Проверка на соответствие гоночной машине | - |

1. **Описание структуры пользовательского меню**

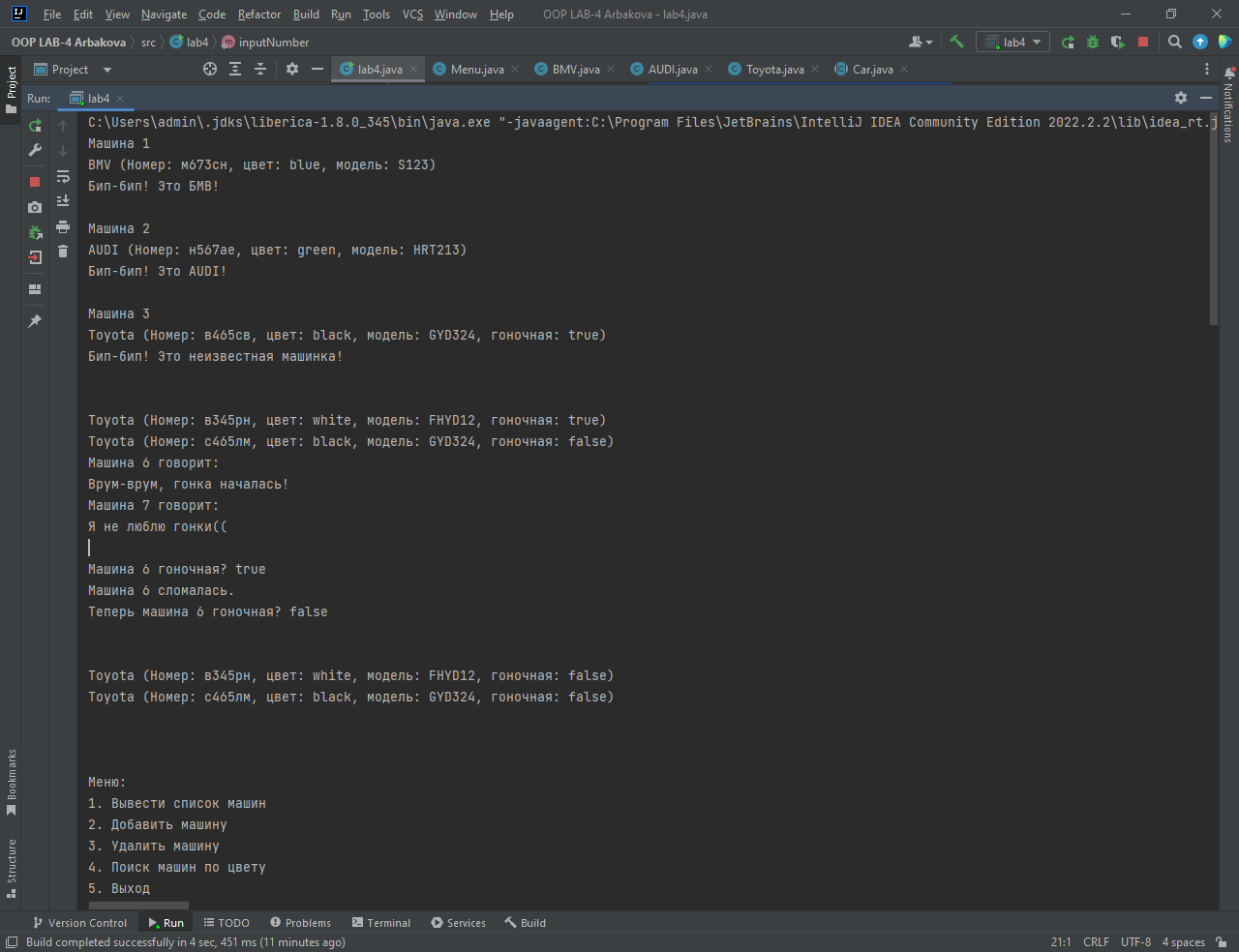
Главное меню программы, состоит из следующих предложенных пользователю вариантов:

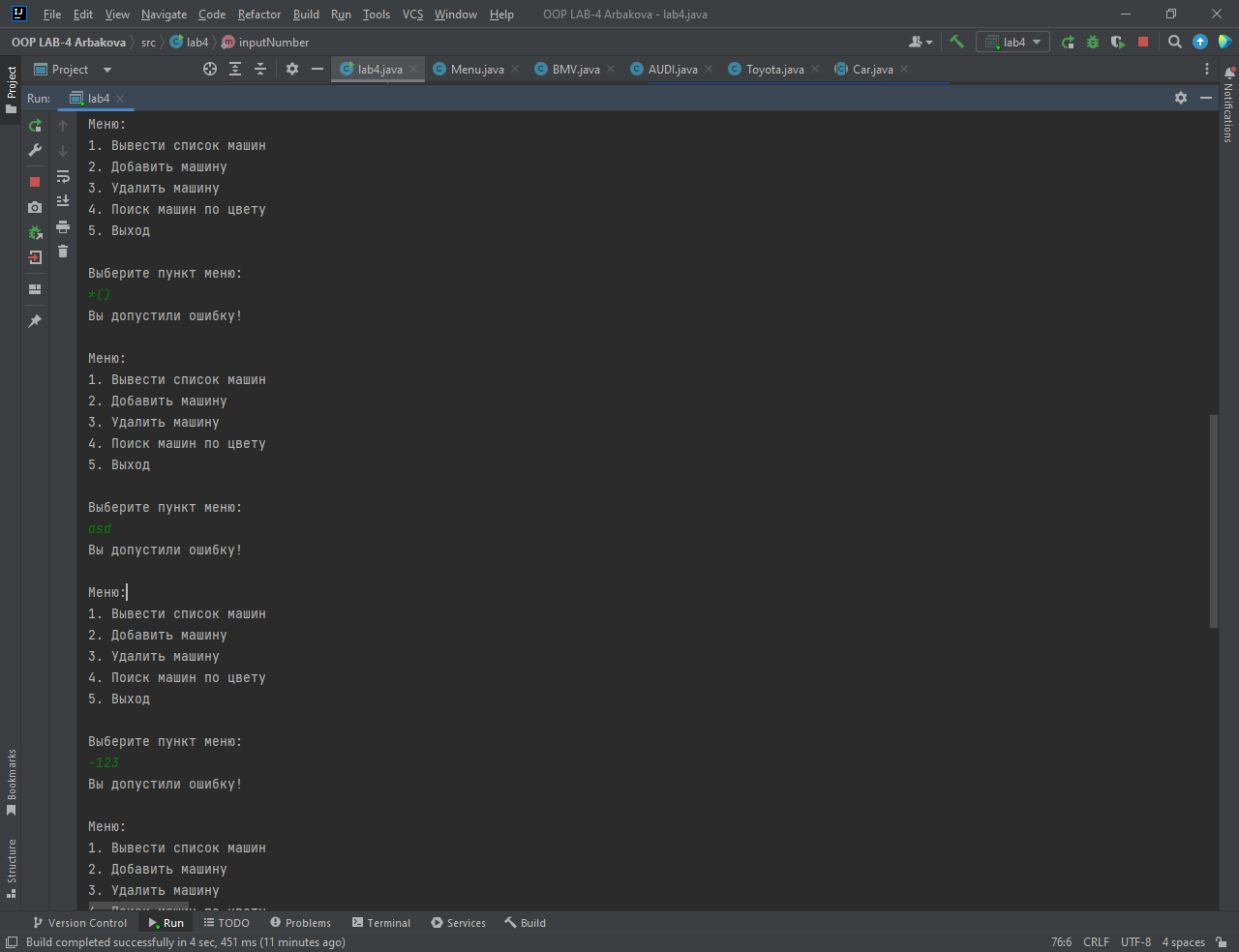
1. Вывести список машин – Осуществляет вывод списка машин.
2. Добавить машину – Осуществляет добавление машины в список.
3. Удалить машину – Осуществляет удаление машины из списка.
4. Поиск машин по цвету – Осуществляет поиск по цвету машины из списка.
5. Выход – Выполняется выход из программы.
6. **Таблица тестов**

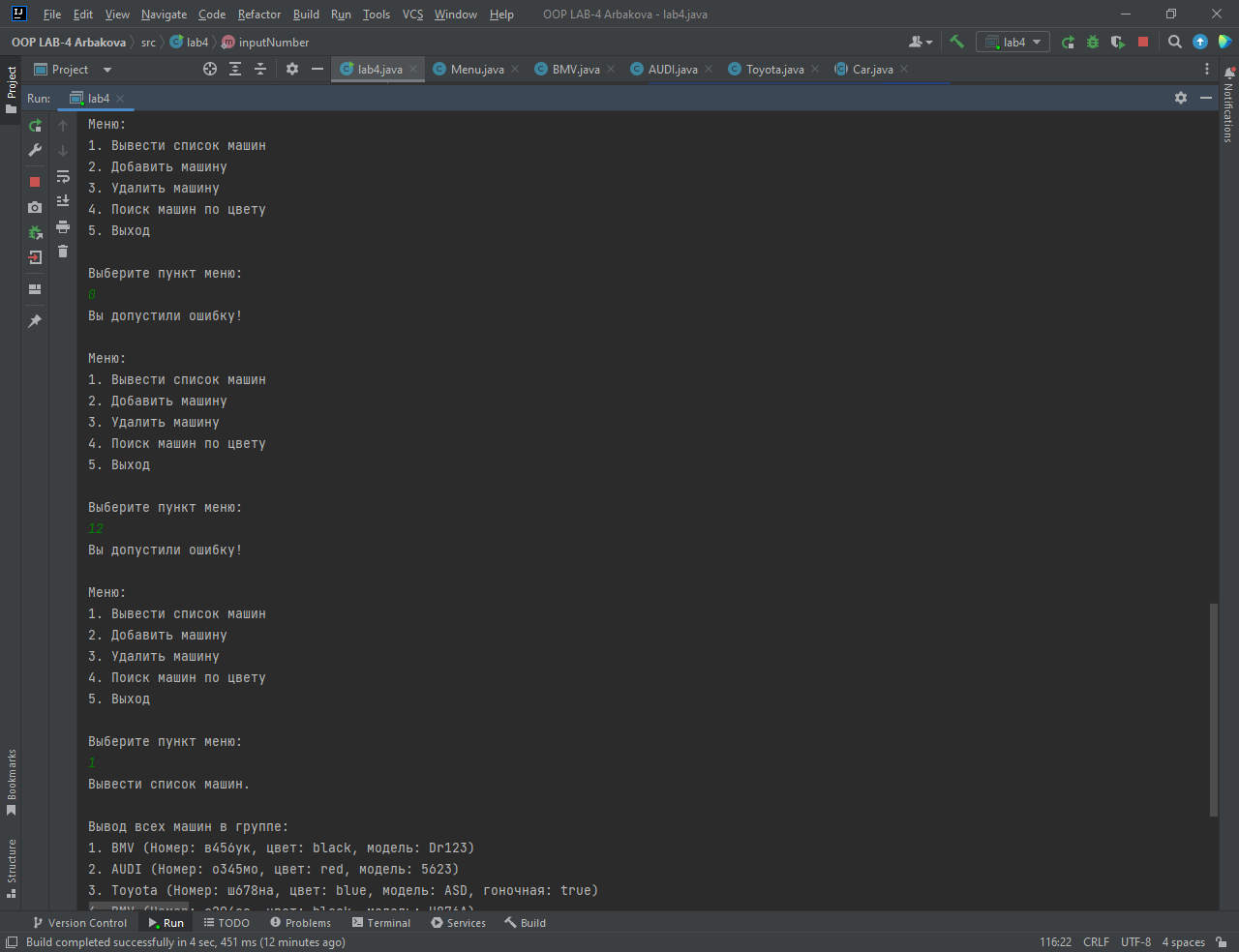
|  |  |
| --- | --- |
| **Исходные данные** | |
| Машина 1 (car3) | BMV (Номер: м673сн, цвет: blue, модель: S123) |
| Машина 2 (car4) | AUDI (Номер: н567ае, цвет: green, модель: HRT213) |
| Машина 3 (car5) | Toyota (Номер: в465св, цвет: black, модель: GYD324, гоночная: true) |
| Машина 4 (car6) | Toyota (Номер: в345рн, цвет: white, модель: FHYD12, гоночная: true) |
| Машина 5 (car7) | Toyota (Номер: с465лм, цвет: black, модель: GYD324, гоночная: false) |
| car1 | BMV (Номер: в456ук, цвет: black, модель: Dr123) |
| car2 | AUDI (Номер: о345мо, цвет: red, модель: 5623) |
| car3 | Toyota (Номер: ш678на, цвет: blue, модель: ASD, гоночная: true) |
| car4 | BMV (Номер: с294ае, цвет: black, модель: U876A) |
| car5 | Toyota (Номер: а340ап, цвет: white, модель: 234VD, гоночная: false) |
| car6 | Toyota (Номер: с674ма, цвет: white, модель: 12CS, гоночная: true) |
| car7 | Toyota (Номер: с445св, цвет: white, модель: CCYDJ, гоночная: false) |

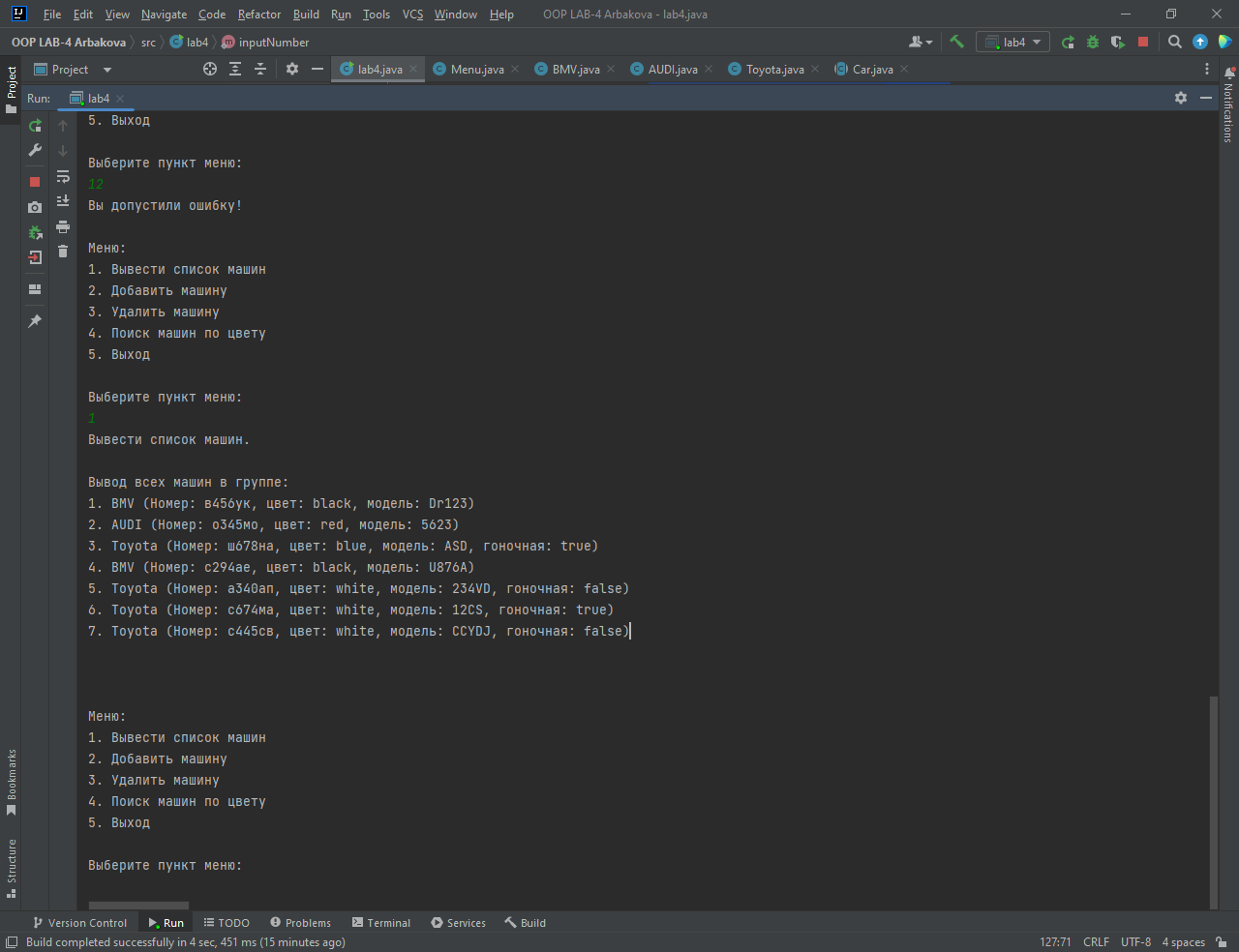
|  |  |
| --- | --- |
| **Результат** | **Примечания** |
| Машина 1  BMV (Номер: м673сн, цвет: blue, модель: S123)  Бип-бип! Это БМВ!  Машина 2  AUDI (Номер: н567ае, цвет: green, модель: HRT213)  Бип-бип! Это AUDI!  Машина 3  Toyota (Номер: в465св, цвет: black, модель: GYD324, гоночная: true)  Бип-бип! Это неизвестная машинка! | Пример переопределения метода beep, отвечающий за сообщение об издаваемом звуке машины. У машин БМВ и AUDI имеется переопределение, а у машины Toyota – нет. |
| Toyota (Номер: в345рн, цвет: white, модель: FHYD12, гоночная: true)  Toyota (Номер: с465лм, цвет: black, модель: GYD324, гоночная: false)  Машина 6 говорит:  Врум-врум, гонка началась!  Машина 7 говорит:  Я не люблю гонки((  Машина 6 гоночная? true  Машина 6 сломалась.  Теперь машина 6 гоночная? false  Toyota (Номер: в345рн, цвет: white, модель: FHYD12, гоночная: false)  Toyota (Номер: с465лм, цвет: black, модель: GYD324, гоночная: false) | Пример изменения переменной isRaceCar, описывающей характеристику гоночной машины. |
| Меню:  1. Вывести список машин  2. Добавить машину  3. Удалить машину  4. Поиск машин по цвету  5. Выход  Выберите пункт меню:  \*()  Вы допустили ошибку!  Выберите пункт меню:  asd  Вы допустили ошибку!  Выберите пункт меню:  -123  Вы допустили ошибку!  Выберите пункт меню:  0  Вы допустили ошибку!  Выберите пункт меню:  12  Вы допустили ошибку!  Выберите пункт меню:  1 | Вывод меню.  Проверка на некорректность ввода:   1. Символы 2. Буквы 3. Отрицательные числа 4. Нуль 5. Число, вне диапазона   Корректный ввод. |
| Вывести список машин.  Вывод всех машин в группе:  1. BMV (Номер: в456ук, цвет: black, модель: Dr123)  2. AUDI (Номер: о345мо, цвет: red, модель: 5623)  3. Toyota (Номер: ш678на, цвет: blue, модель: ASD, гоночная: true)  4. BMV (Номер: с294ае, цвет: black, модель: U876A)  5. Toyota (Номер: а340ап, цвет: white, модель: 234VD, гоночная: false)  6. Toyota (Номер: с674ма, цвет: white, модель: 12CS, гоночная: true)  7. Toyota (Номер: с445св, цвет: white, модель: CCYDJ, гоночная: false) | Вывод списка машин. |
| Выберите пункт меню:  2  Добавить машину.  Введите марку машины ( 1 - БМВ, 2 - Ауди, 3 - Тойота): 1  Введите номер машины:  в567ук  Введите цвет машины:  white  Введите модель машины:  RTY456  Введенная машина: BMV (Номер: в567ук, цвет: white, модель: RTY456)  Вы добавили машину! | Пример добавления в список новой машины. |
| Выберите пункт меню:  3  Удалить машину.  Введите номер из списка:  Введите число от 1 до 8: 4  Удаление машины из группы: BMV (Номер: с294ае, цвет: black, модель: U876A) | Пример удаления машины из группы. |
| Выберите пункт меню:  4  Поиск машин по цветам.  Введите цвет:  white  Найденные машины:  Toyota (Номер: а340ап, цвет: white, модель: 234VD, гоночная: false)  Toyota (Номер: с674ма, цвет: white, модель: 12CS, гоночная: true)  Toyota (Номер: с445св, цвет: white, модель: CCYDJ, гоночная: false)  BMV (Номер: в567ук, цвет: white, модель: RTY456) | Пример поиска машины по цвету. |
| Меню:  1. Вывести список машин  2. Добавить машину  3. Удалить машину  4. Поиск машин по цвету  5. Выход  Выберите пункт меню:  5  Выполнен выход из программы. | Выход из программы. |

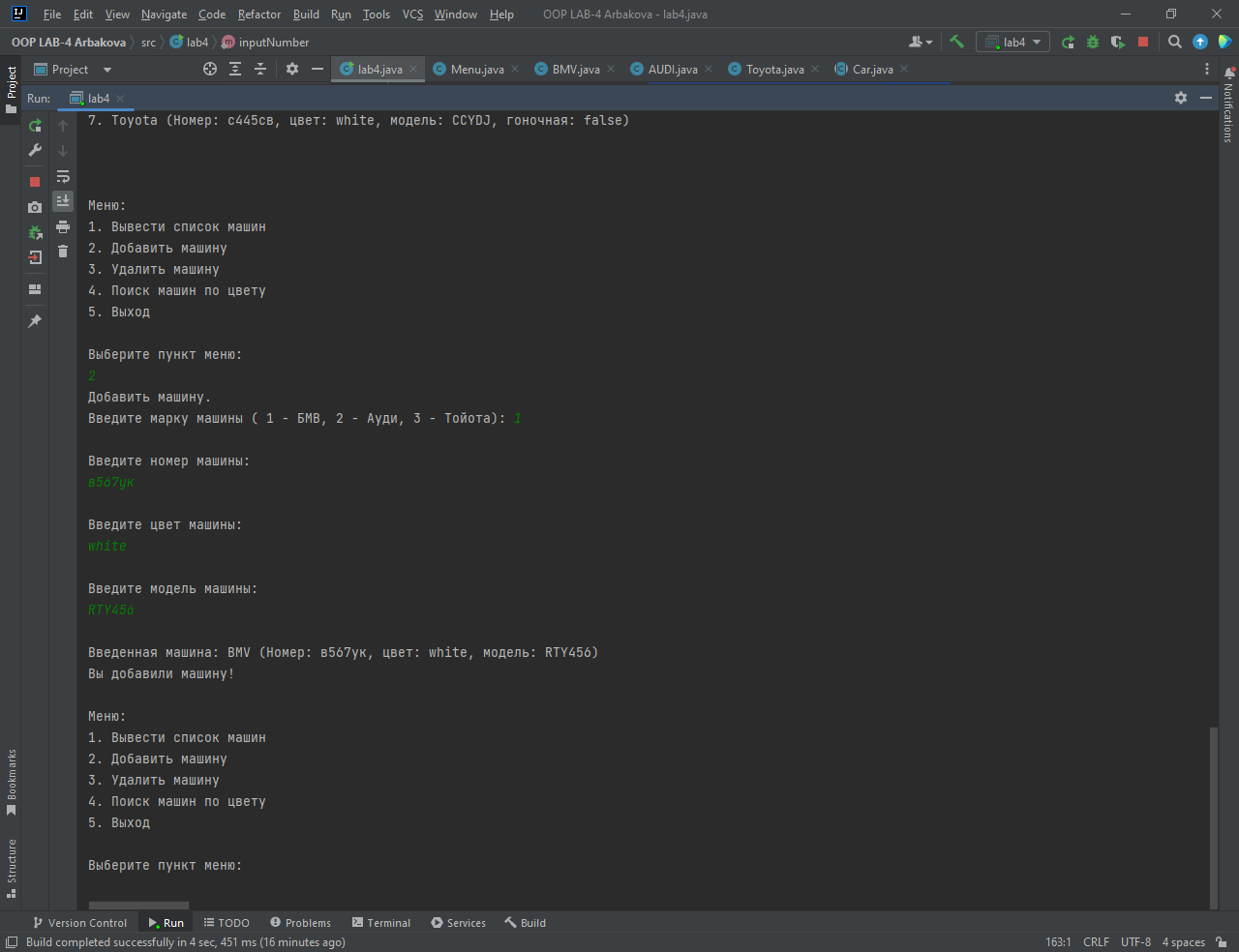
1. **Результаты тестирования**

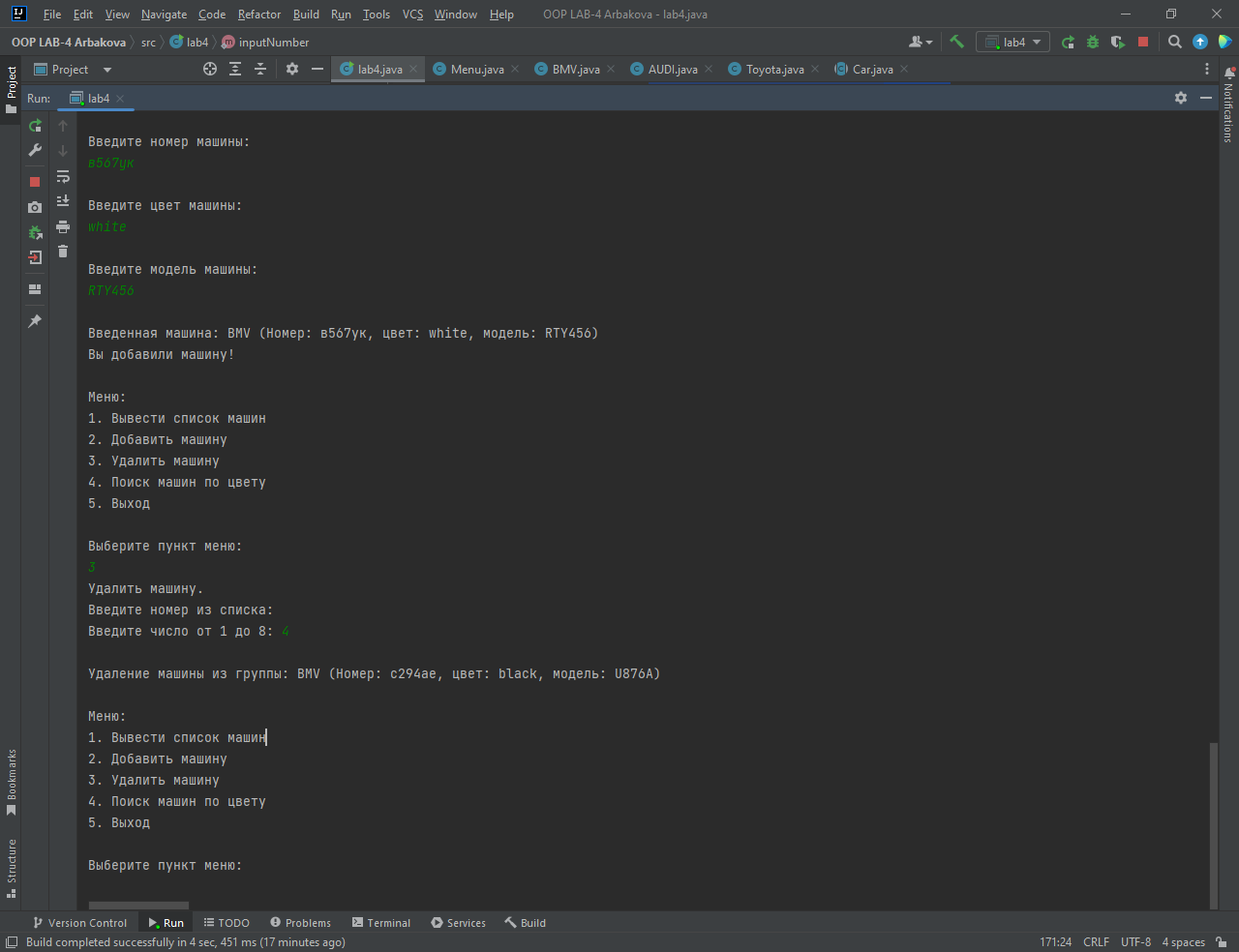


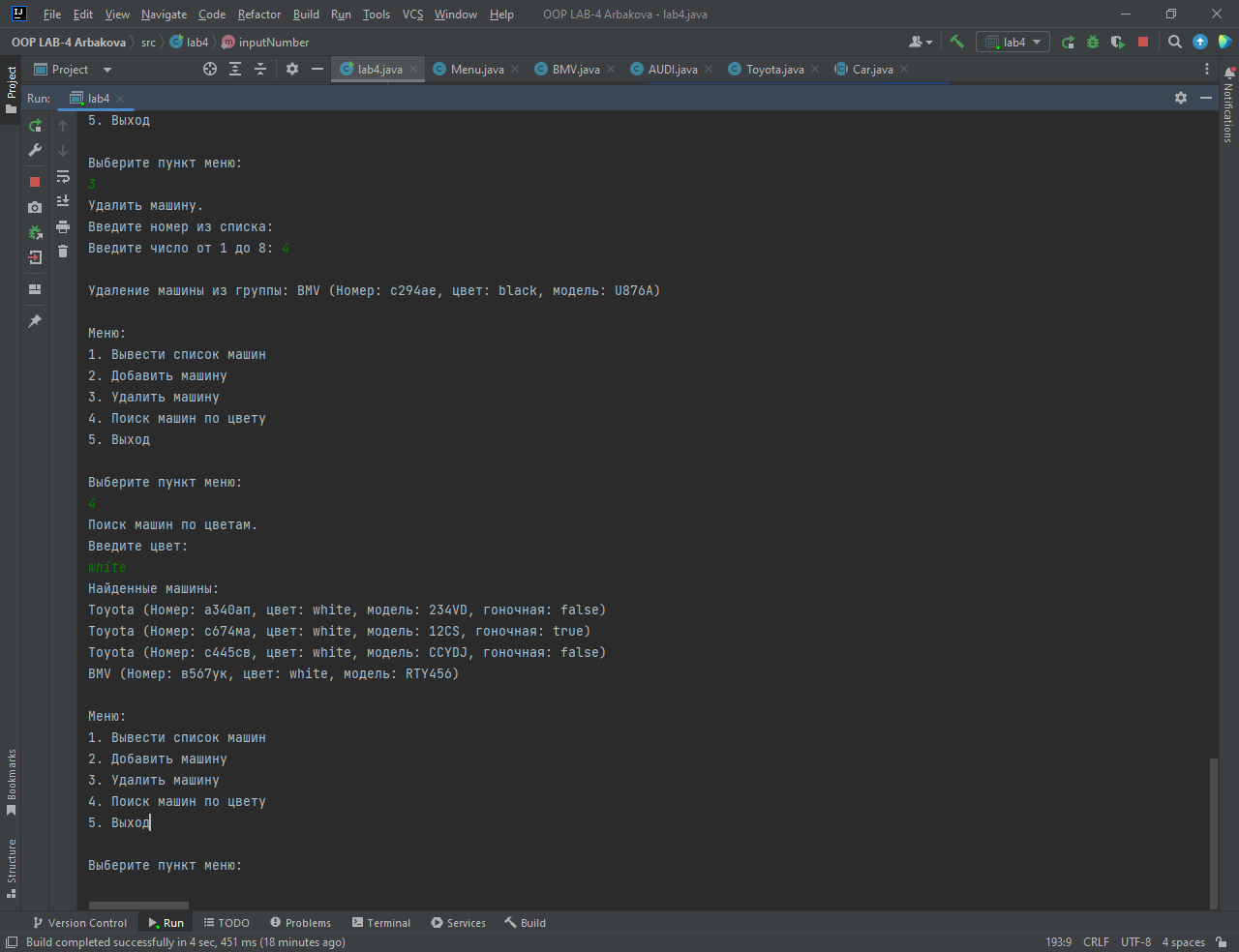


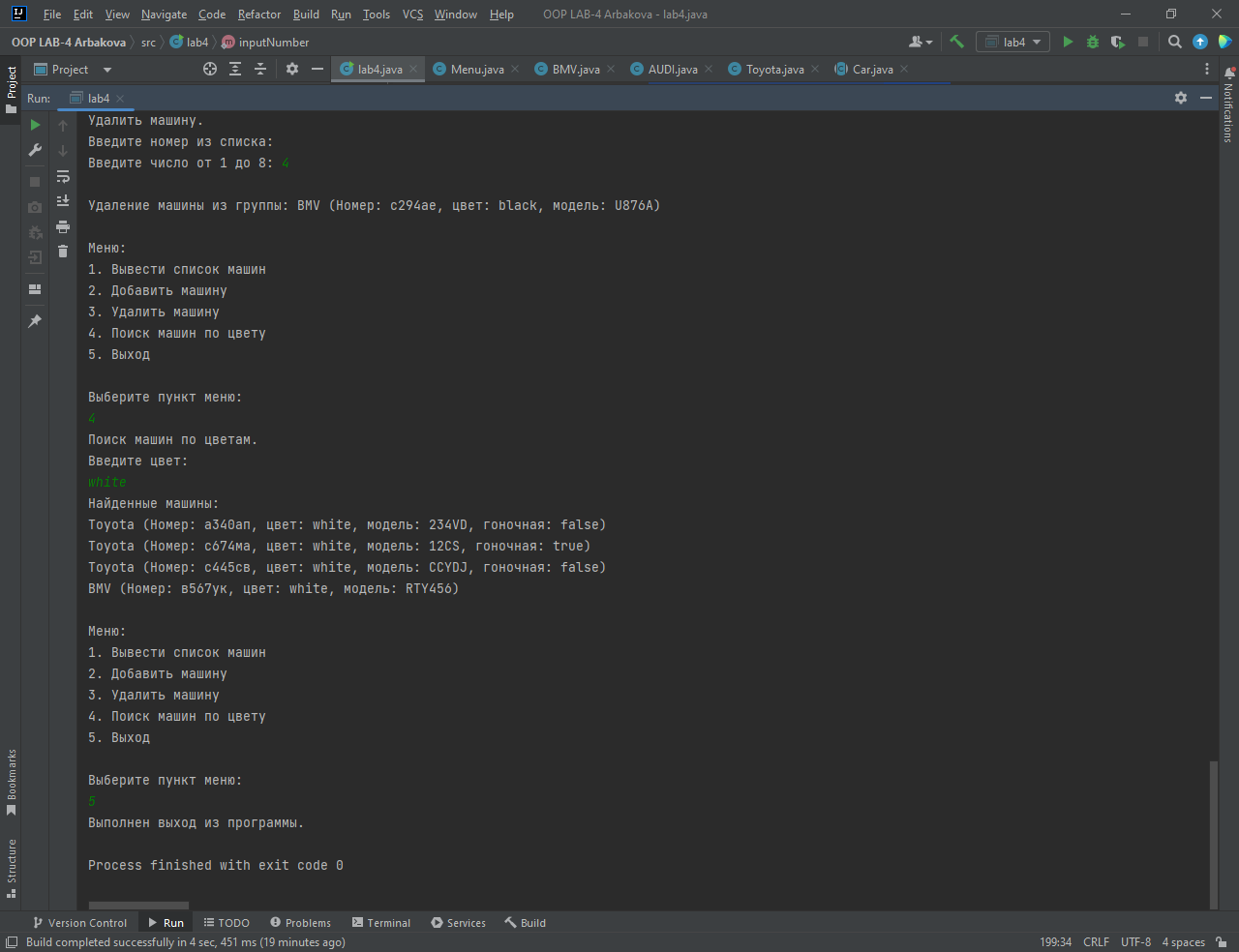












1. **Исходный код**

Класс lab4:

import java.util.Scanner;  
  
public class lab4 {  
 public static void main(String[] args) {  
 //пример наследования, полиморфизма, инкапсуляции  
 Car car3 = new BMV("м673сн", "blue", "S123"); //присвоение ссылки абстрактного класса - класса наследника (car-абстрактный класс bmv-наследник)  
 Car car4 = new AUDI("н567ае", "green", "HRT213");  
 Car car5 = new Toyota("в465св", "black", "GYD324", true); //все 3 машины - пример полиформизма  
 //абстрактный класс имя = new класс-наследник  
 //пример переопределения  
 System.*out*.println("Машина 1");  
 System.*out*.println(car3);  
 car3.beep();  
 System.*out*.println("\nМашина 2");  
 System.*out*.println(car4);  
 car4.beep();  
 System.*out*.println("\nМашина 3");  
 System.*out*.println(car5);  
 car5.beep();  
 //пример на машинах 6 и 7  
 System.*out*.println("\n");  
 Toyota car6 = new Toyota("в345рн", "white", "FHYD12", true);  
 Toyota car7 = new Toyota("с465лм", "black", "GYD324", false);  
 System.*out*.println(car6);  
 System.*out*.println(car7);  
 System.*out*.println("Машина 6 говорит: ");  
 car6.startRace();  
 System.*out*.println("Машина 7 говорит: ");  
 car7.startRace();  
 //замена значения переменной isRaceCar  
 System.*out*.println("\nМашина 6 гоночная? "+car6.isRaceCar());  
 System.*out*.println("Машина 6 сломалась.");  
 car6.setRaceCar(false);  
 System.*out*.println("Теперь машина 6 гоночная? "+car6.isRaceCar());  
 System.*out*.println("\n");  
 System.*out*.println(car6);  
 System.*out*.println(car7);  
 System.*out*.println("\n");  
  
 Menu.*menu*();  
 }  
  
 public static int inputNumber(int leftGranitcha, int rightGranitcha) { //функция ввода размерности матрицы  
 String input = "";  
 int x = 0;  
 boolean isCorrect = true;  
 Scanner console = new Scanner(System.*in*);  
 do {  
 try {  
 System.*out*.print("Введите число от " + leftGranitcha + " до " + (rightGranitcha-1) + ": ");  
 input = console.nextLine();  
 x = Integer.*parseInt*(input);  
 if ((x>=leftGranitcha) && (x <= rightGranitcha)){  
 break;  
 } else { System.*out*.print("\nНекорректный ввод!\n");}  
 }catch(NumberFormatException e) {  
 isCorrect = false;  
 System.*out*.print("\nОшибка! Введите число!\n");  
 }  
 } while (true);  
 return x;  
 }  
}

Класс Menu:

import java.util.ArrayList;  
import java.util.List;  
import java.util.Scanner;  
  
public class Menu {  
 public static void menu (){  
 StringBuilder builder = new StringBuilder();  
 builder.append("\nМеню:\n") //пункты меню  
 .append("1. Вывести список машин\n")  
 .append("2. Добавить машину\n")  
 .append("3. Удалить машину\n")  
 .append("4. Поиск машин по цвету\n")  
 .append("5. Выход\n");  
 boolean isWork = true;  
 List<Car> carList = new ArrayList<>(); //список машин  
 String number;  
 String color;  
 String model;  
 String raceCar; boolean isRaceCar = false;  
 Scanner scanner = new Scanner(System.*in*);  
 Car car1 = new BMV("в456ук", "black", "Dr123");  
 Car car2 = new AUDI("о345мо", "red", "5623");  
 Car car3 = new Toyota("ш678на", "blue", "ASD", true);  
 Car car4 = new BMV("с294ае", "black", "U876A");  
 Car car5 = new Toyota("а340ап", "white", "234VD", false);  
 Car car6 = new Toyota("с674ма", "white", "12CS", true);  
 Car car7 = new Toyota("с445св", "white", "CCYDJ", false);  
 carList.add(car1); carList.add(car2); carList.add(car3); carList.add(car4); carList.add(car5); carList.add(car6); carList.add(car7);  
 while (isWork) {  
 System.*out*.println(builder.toString()); //вывод меню на экран  
 System.*out*.print("Выберите пункт меню: \n"); String i = new Scanner(System.*in*).nextLine();  
 switch (i) {  
 case "1":  
 System.*out*.print("Вывести список машин.\n");  
 System.*out*.println("\nВывод всех машин в группе:");  
 for(int j = 0; j < carList.size(); j++) {  
 System.*out*.print(j+1);  
 System.*out*.print(". ");  
 System.*out*.println(carList.get(j));  
 }  
 System.*out*.println("\n");  
 break;  
  
 case "2":  
 System.*out*.print("Добавить машину.\n");  
 System.*out*.print("Введите марку машины ( 1 - БМВ, 2 - Ауди, 3 - Тойота): ");  
 String marka = scanner.nextLine();  
 Car car;  
 switch (marka) {  
 case "1":  
 System.*out*.println("\nВведите номер машины: ");  
 number = new Scanner(System.*in*).nextLine();  
 System.*out*.println("\nВведите цвет машины: ");  
 color = new Scanner(System.*in*).nextLine();  
 System.*out*.println("\nВведите модель машины: ");  
 model = new Scanner(System.*in*).nextLine();  
 car = new BMV(number, color, model);  
 System.*out*.println("\nВведенная машина: " + car);  
 carList.add(car);  
 System.*out*.print("Вы добавили машину!\n");  
 break;  
  
 case "2":  
 System.*out*.println("\nВведите номер машины: ");  
 number = new Scanner(System.*in*).nextLine();  
 System.*out*.println("\nВведите цвет машины: ");  
 color = new Scanner(System.*in*).nextLine();  
 System.*out*.println("\nВведите модель машины: ");  
 model = new Scanner(System.*in*).nextLine();  
 car = new AUDI(number, color, model);  
 System.*out*.println("\nВведенная машина: " + car);  
 carList.add(car);  
 System.*out*.print("Вы добавили машину!\n");  
 break;  
  
 case "3":  
 System.*out*.println("\nВведите номер машины: ");  
 number = new Scanner(System.*in*).nextLine();  
 System.*out*.println("\nВведите цвет машины: ");  
 color = new Scanner(System.*in*).nextLine();  
 System.*out*.println("\nВведите модель машины: ");  
 model = new Scanner(System.*in*).nextLine();  
 System.*out*.println("\nВведите является ли эта машина гоночной (Да/Нет): ");  
 raceCar = new Scanner(System.*in*).nextLine();  
  
 if ((raceCar.equals("Да")) || (raceCar.equals("да")) || (raceCar.equals("ДА"))) {  
 isRaceCar = true;  
 } else if ((raceCar.equals("Нет")) || (raceCar.equals("нет")) || (raceCar.equals("НЕТ"))) {  
 isRaceCar = false;  
 } else System.*out*.println("\nВведите Да или Нет. Попробуйте еще раз!");  
  
 car = new Toyota(number, color, model, isRaceCar);  
 System.*out*.println("\nВведенная машина: " + car);  
 carList.add(car);  
 System.*out*.print("Вы добавили машину!\n");  
 break;  
  
 default:  
 System.*out*.println("Введите 1 - БМВ, 2 - Ауди или 3 - Тойота!");  
 }  
 break;  
 case "3":  
 System.*out*.print("Удалить машину.\n");  
 System.*out*.print("Введите номер из списка: \n");  
 int input = lab4.*inputNumber*(1, carList.size()+1);  
 if (input > carList.size()){  
 System.*out*.println("Введите число от 1 до "+ (carList.size()+1) + "!");  
 } else  
 System.*out*.println("\nУдаление машины из группы: "+carList.remove(input-1));  
 break;  
  
 case "4":  
 System.*out*.print("Поиск машин по цветам.\n");  
 System.*out*.println("Введите цвет: ");  
 String colorSearch = new Scanner(System.*in*).nextLine();  
 System.*out*.println("Найденные машины: ");  
 for (Car carInList: carList){  
 if (carInList.getColor().equals(colorSearch)) {  
 System.*out*.println(carInList);  
 }  
 }  
 break;  
  
 case "5":  
 System.*out*.print("Выполнен выход из программы.\n"); //выход из программы  
 isWork = false;  
 break;  
  
 default:  
 System.*out*.println("Вы допустили ошибку!");  
 }  
 }  
 }  
}

Класс Car:

public abstract class Car { //абстрактный класс машина  
 private String number;  
 private String color;  
 private String model;  
  
 public Car(String number, String color, String model) { //конструктор  
 this.number = number;  
 this.color = color;  
 this.model = model;  
 }  
  
 @Override  
 public String toString() {  
 String[] classInfo = this.getClass().toString().split(" ");  
 return classInfo[1]+ " (Номер: "+number+", цвет: "+color+", модель: "+model+")";  
 }  
  
 public void beep(){  
 System.*out*.println("Бип-бип! Это неизвестная машинка!");  
 }  
  
 //getters setters  
 public String getNumber() {  
 return number;  
 }  
  
 public void setNumber(String number) {  
 this.number = number;  
 }  
  
 public String getColor() {  
 return color;  
 }  
  
 public void setColor(String color) {  
 this.color = color;  
 }  
  
 public String getModel() {  
 return model;  
 }  
  
 public void setModel(String model) {  
 this.model = model;  
 }  
}

Класс BMV:

public class BMV extends Car { //бмв наследник класса машина  
 public BMV(String number, String color, String model) {  
 super(number, color, model); //супер конструктор, который уже находится в абстрактном классе  
 }  
 @Override //перезапись метода  
 public void beep(){  
 System.*out*.println("Бип-бип! Это БМВ!");  
 }  
  
}

Класс AUDI:

public class AUDI extends Car { //ауди наследник класса машина  
 public AUDI(String number, String color, String model) {  
 super(number, color, model); //супер конструктор, который уже находится в абстрактном классе  
 }  
 @Override //перезапись метода  
 public void beep(){  
 System.*out*.println("Бип-бип! Это AUDI!");  
 }  
}

Класс Toyota:

public class Toyota extends Car {  
 private boolean isRaceCar;  
  
 public Toyota(String number, String color, String model, boolean isRaceCar) {  
 super(number, color, model); //супер конструктор, который уже находится в абстрактном классе  
 this.isRaceCar = isRaceCar;  
 }  
 public void startRace(){  
 if (this.isRaceCar)  
 System.*out*.println("Врум-врум, гонка началась!");  
 else  
 System.*out*.println("Я не люблю гонки((");  
 }  
 @Override  
 public String toString() {  
 String[] classInfo = this.getClass().toString().split(" ");  
 return classInfo[1]+ " (Номер: "+this.getNumber()+", цвет: "+this.getColor()+", модель: "+this.getModel()+", гоночная: "+this.isRaceCar+")";  
 }  
 public void setRaceCar(boolean raceCar) { //setter - установка значения  
 isRaceCar = raceCar;  
 }  
 public boolean isRaceCar() { //getter - возвращает переменную  
 return isRaceCar;  
 }  
}

1. **Список использованных источников**
2. Шилдт, Герберт Java 8. Руководство для начинающих / Герберт Шилдт. - М.: Вильямс, 2015. - 720 c.
3. Берд, Барри Java для чайников / Барри Берд. - М.: Диалектика / Вильямс, 2013. - 521 c.
4. Савитч, Уолтер Язык Java. Курс программирования / Уолтер Савитч. - М.: Вильямс, 2015. - 928 c.