Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

**(ВлГУ)**

**Кафедра информационных систем и программной инженерии**

Лабораторная работа №2

по дисциплине «Платформонезависимое

программирование»

Тема: «Интерфейсы. Наследование.»

Выполнил:

студент гр. ПРИ-123

А.Ц. Нямаа

Приняла:

Ст. пр. кафедры ИСПИ

О.Н. Шамышева

Владимир, 2025 г.

ЦЕЛЬ РАБОТЫ:

Освоить базовые принципы ООП. Получить практические навыки по работе с классами и объектами на платформе Java. Научиться использовать наследование, абстрактные классы, интерфейсы.

ЗАДАНИЕ 1 (вариант 12).

Выполнить и доработать пример создания приложения с абстрактным классом и интерфейсом согласно методическому указанию.

ВЫПОЛНЕНИЕ РАБОТЫ:

В начале выполнения данного задания я определил физическую структуру всего приложения.

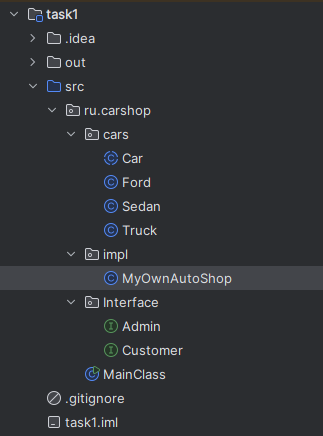


Рисунок 1. Физическая структура приложения.

Далее в пакете carshop.cars создал абстрактный класс Car с некоторыми заданными в методичке полями и методами. Также создал 3 расширяющих класса Truck, Ford и Sedan. Следующим шагом в пакете carshop.interface создал два интерфейса в Admin и Customer. Реализовал данные интерфейсы в классе MyOwnAutoShop и создал 2 объекта Sedan, 2 объекта Ford и 2 объекта Truck. Продемонстрировал работу приложения в статическом классе main.

Листинг основного класса MainClass.java:

package ru.carshop;  
  
import ru.carshop.impl.MyOwnAutoShop;  
import ru.carshop.cars.Car;  
  
import java.util.Scanner;  
  
public class MainClass {  
 public static void main(String [] args){  
 Scanner in = new Scanner(System.in);  
 MyOwnAutoShop newOne = new MyOwnAutoShop();  
  
 while (true){  
 System.out.println("-----------------------------");  
 System.out.println("Выберите команду:\n1 - Напечатать цены всех авто\n2 - Напечатать цвета всех авто\n" +  
 "3 - Напечатать цену авто под id\n4 - Напечатать цвет авто под id\n5 - Выбрать авто для покупки\n" +  
 "6 - Узнать сумму всех покупок за сегодня\n7 - Завершить программу");  
 int choice = in.nextInt();  
 System.out.println();  
  
 if (choice == 1){  
 double[] priceList = newOne.getCarsPrice();  
 for(int i = 0; i < priceList.length; i++){  
 System.out.println("Цена авто под Id - " + i + ": " + priceList[i]);  
 }  
 }  
 else if(choice == 2){  
 String[] colorList = newOne.getCarsColors();  
 for(int i = 0; i < colorList.length; i++){  
 System.out.println("Цвет авто под Id - " + i + ": " + colorList[i]);  
 }  
 }  
 else if(choice == 3){  
 System.out.println("Введите нужное айди");  
 int choiceID = in.nextInt();  
 System.out.println("Цена авто под Id - " + choiceID + ": " + newOne.getCarPrice(choiceID));  
 }  
 else if(choice == 4){  
 System.out.println("Введите нужное айди");  
 int choiceID = in.nextInt();  
 System.out.println("Цвет авто под Id - " + choiceID + ": " + newOne.getCarColor(choiceID));  
 }  
 else if(choice == 5){  
 System.out.println("Введите нужное айди");  
 int choiceID = in.nextInt();  
 System.out.println("Поздравляю! Вы купили машину под Id - " + choiceID + " и имеющую следующие "  
 + "характеристики: цена со скидкой - " + newOne.getCarPrice(choiceID) + ", "  
 + newOne.purchaseCar(choiceID).giveInfoCar());  
 }  
 else if(choice == 6){  
 System.out.println("Сумма всех купленных за сессию авто: " + newOne.getIncome());  
 }  
 else if(choice == 7){  
 break;  
 }  
 }  
 }  
}

Листинг абстрактного класса Car.java

package ru.carshop.cars;  
  
public abstract class Car {  
 int speed;  
 boolean isSellOut;  
 double regularPrice;  
 public String color;  
  
 public Car(int speed, boolean isSellOut, double regularPrice, String color){  
 this.speed = speed;  
 this.isSellOut = isSellOut;  
 this.regularPrice = regularPrice;  
 this.color = color;  
 }  
  
 public abstract double getSalePrice();  
  
 public String giveInfoCar(){  
 return "скорость машины - " + speed + ", обычная цена авто - " + regularPrice + ", цвет авто - " + color;  
 }  
}

Листинг расширяющего класса Ford.java

package ru.carshop.cars;  
  
public class Ford extends Car{  
 int year;  
 double manufacturerDiscount;  
  
 public Ford(int speed, boolean isSeelOut, int regularPrice, String color, int year, double manufacturerDiscount) {  
 super(speed, isSeelOut, regularPrice, color);  
 this.year = year;  
 this.manufacturerDiscount = manufacturerDiscount;  
 }  
  
 @Override  
 public double getSalePrice() {  
 return regularPrice - (regularPrice \* manufacturerDiscount);  
 }  
}

Листинг расширяющего класса Sedan.java

package ru.carshop.cars;  
  
public class Sedan extends Car{  
 int length;  
  
 public Sedan(int speed, boolean isSeelOut, int regularPrice, String color, int length) {  
 super(speed, isSeelOut, regularPrice, color);  
 this.length = length;  
 }  
  
 @Override  
 public double getSalePrice() {  
 if (length > 20){  
 return regularPrice \* 0.95;  
 }  
 return regularPrice;  
 }  
}

Листинг расширяющего класса Truck.java

package ru.carshop.cars;  
  
public class Truck extends Car {  
 int weight;  
  
 public Truck(int speed, boolean isSeelOut, int regularPrice, String color, int weight) {  
 super(speed, isSeelOut, regularPrice, color);  
 this.weight = weight;  
 }  
  
 @Override  
 public double getSalePrice() {  
 if (weight > 2000){  
 return regularPrice \* 0.9;  
 }  
 return regularPrice;  
 }  
}

Листинг интерфейса Admin.java

package ru.carshop.Interface;  
  
public interface Admin {  
 double getIncome();  
}

Листинг интерфейса Customer.java

package ru.carshop.Interface;  
  
import ru.carshop.cars.Car;  
  
public interface Customer {  
 double[] getCarsPrice();  
 String[] getCarsColors();  
 double getCarPrice(int id);  
 String getCarColor(int id);  
 Car purchaseCar(int id);  
}

Листинг реализующего класса MyOwnAutoShop.java

package ru.carshop.impl;  
  
import ru.carshop.cars.\*;  
import ru.carshop.Interface.\*;  
  
import java.util.ArrayList;  
  
public class MyOwnAutoShop implements Admin, Customer{  
  
 private ArrayList<Car> cars;  
 private double allValue = 0;  
  
 public MyOwnAutoShop(){  
 cars = new ArrayList<>();  
  
 cars.add(new Sedan(125, false, 14000, "White", 200));  
 cars.add(new Ford(250, false, 25000, "Black", 2022, 0.15));  
 cars.add(new Ford(200, false, 30000, "Red", 2025, 0.15));  
 cars.add(new Truck(375, false, 22500, "Black", 2000));  
 cars.add(new Truck(300, false, 50000, "Purple", 3000));  
 }  
  
 @Override  
 public double getIncome()  
 {  
 return allValue;  
 }  
  
 @Override  
 public double[] getCarsPrice() {  
 double[] priceList = new double[cars.size()];  
 for (int i = 0; i < cars.size(); i++){  
 priceList[i] = cars.get(i).getSalePrice();  
 }  
 return priceList;  
 }  
  
 @Override  
 public String[] getCarsColors() {  
 String[] colors = new String[cars.size()];  
 for (int i = 0; i < cars.size(); i++){  
 colors[i] = cars.get(i).color;  
 }  
 return colors;  
 }  
  
 @Override  
 public double getCarPrice(int id) {  
 return cars.get(id).getSalePrice();  
 }  
  
 @Override  
 public String getCarColor(int id) {  
 return cars.get(id).color;  
 }  
  
 @Override  
 public Car purchaseCar(int id) {  
 Car itemCar = cars.get(id);  
 cars.remove(id);  
 allValue += itemCar.getSalePrice();  
 return itemCar;  
 }  
}

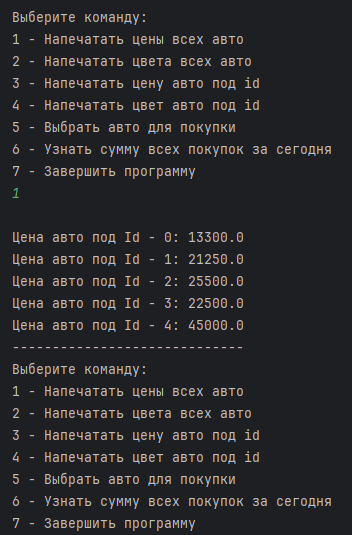


Рисунок 2. Выполнение программы. Часть 1.

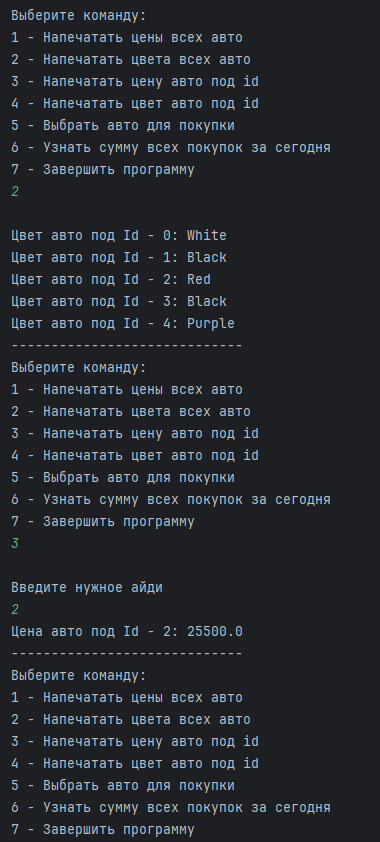


Рисунок 3. Выполнение программы. Часть 2.

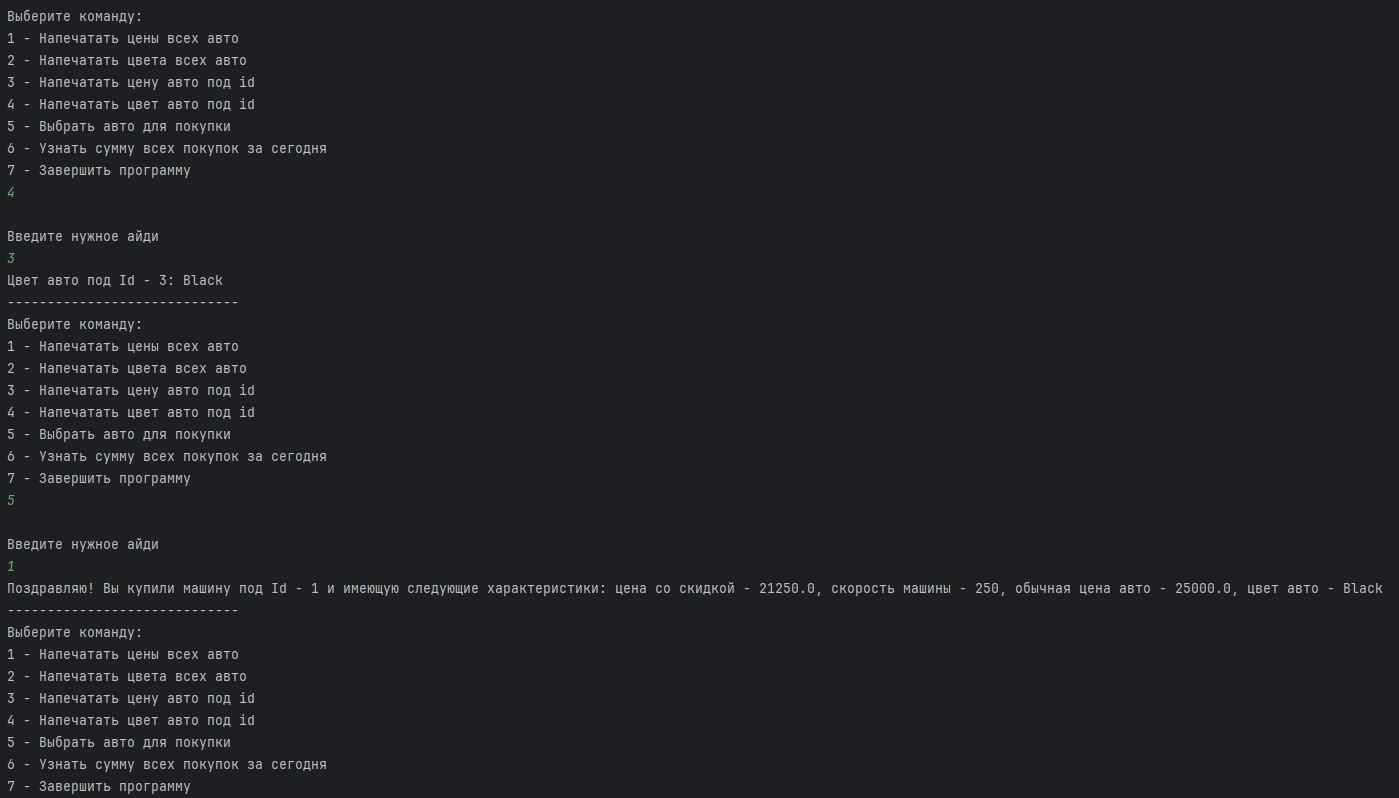


Рисунок 4. Выполнение программы. Часть 3.

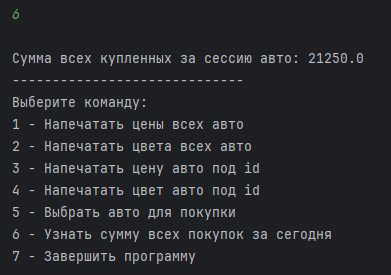


Рисунок 5. Выполнение программы. Часть 4.

ЗАДАНИЕ 2 (вариант 12).

Необходимо описать интерфейс, содержащий одну функцию *int count(int number)*. Данный интерфейс позволяет вычислить характеристики чисел. Необходимо реализовать два класса для этого интерфейса. Первый класс должен вычислять количество символов в десятичной записи этого числа. Второй класс должен вычислять количество различных простых чисел, являющихся множителями данного числа. Необходимо, чтобы приложение заспросило у пользователя число и выдало статистические сведения для обоих классов.

ВЫПОЛНЕНИЕ РАБОТЫ:

Для выполнения данного задания я создал физическую структуру.

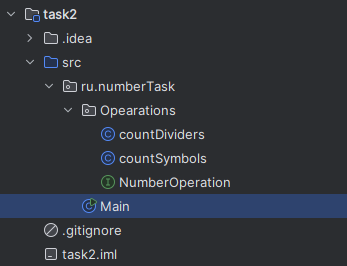


Рисунок 6. Физическая структура программы.

Описал интерфейс, содержащий одну функцию count(), который позволяет вычислить характеристики числа. Реализовал два класса для этого интерфейса, первый из которых вычисляет количество символов в десятичной записи этого числа, а второй класс вычисляет количество различных простых чисел, являющихся множителями данного числа. Данные запрашиваются у пользователя, также выдает статические сведения для обоих классов.

Листинг основного класса Main.java:

package ru.numberTask;  
  
import ru.numberTask.Opearations.\*;  
  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.in);  
  
 System.out.println("Введите цифру");  
 int number = in.nextInt();  
  
 countSymbols countSymbols = new countSymbols();  
 System.out.println("Количество символов в десятичной записи: " + countSymbols.count(Math.abs(number)));  
  
 countDividers countDividers = new countDividers();  
 System.out.println("Количество различных простых чисел, являющихся множителями: " + countDividers.count(Math.abs(number)));  
  
  
 System.out.println("----------------------\nСтатистические данные классов:");  
 System.out.println("Класс countSymbols:\nДата и время создания: " + countSymbols.CreateTime  
 + ", количество запусков: " + countSymbols.countRun);  
 System.out.println("Класс countDividers:\nДата и время создания: " + countDividers.CreateTime  
 + ", количество запусков: " + countDividers.countRun);  
 }  
}

Листинг основного класса NumberOperation.java:

package ru.numberTask.Opearations;  
  
interface NumberOperation {  
 int count(int num);  
}

Листинг основного класса countSymbols.java:

package ru.numberTask.Opearations;  
  
import java.util.Date;  
  
public class countSymbols implements NumberOperation{  
 public Date CreateTime = new Date();  
 public int countRun = 0;  
  
 @Override  
 public int count(int num)  
 {  
 countRun++;  
 return String.valueOf(num).length();  
 }  
}

Листинг основного класса countDividers.java:

package ru.numberTask.Opearations;  
  
import java.util.Date;  
import java.util.ArrayList;  
  
public class countDividers implements NumberOperation{  
 public Date CreateTime = new Date();  
 public int countRun = 0;  
  
 @Override  
 public int count(int num) {  
 ArrayList<Integer> tempList = new ArrayList<>();  
 countRun++;  
  
 for (int divider = 1; divider <= num; divider++){  
 if (num % divider == 0){  
 boolean flag = true;  
  
 for (int checker = 2; Math.pow(checker, 2) <= divider; checker++){  
 if (divider % checker == 0) {  
 flag = false;  
 break;  
 }  
 }  
  
 if (flag) {  
 tempList.add(divider);  
 }  
 }  
 }  
 return tempList.size();  
 }  
}

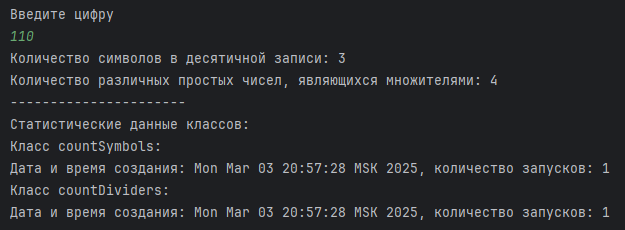


Рисунок 7. Выполнение программы.

ВЫВОД К РАБОТЕ:

Я освоил базовые принципы ООП. Получил практические навыки по работе с классами и объектами на платформе Java. Научился использовать наследование, абстрактные классы, интерфейсы.